Wydział Matematyki, Informatyki i Mechaniki UW

Automaty a logika

skrypt z wykładu i ćwiczeń

semestr zimowy 2009/2010

Wykład prowadził	**dr hab. Mikołaj Bojańczyk**
a czasem go zastępował	**mgr Paweł Parys**
Ćwiczenia prowadzili	**mgr Filip Murlak** **mgr Paweł Parys**
Teksty spisali	**Michał Gołębiowski** **Krzysztof Kąs** **Marek Kiszkis** **Mateusz Kopeć** **Michał Pilipczuk** **Oskar Skibski** **Michał Skrzypczak** **Karolina Sołtys** **Piotr Szczepański** **Mateusz Tarkowski** **Adam Witkowski** **Maciej Zdanowicz**

Spis treści

Rozdział 1

Słowa skończone

Spisali: Karolina Sołtys i Mateusz Tarkowski

1.1 Wstęp

Zacznijmy od obserwacji, że możemy łatwo opisywać pewne własności słów za pomocą formuł logicznych. Załóżmy, że dysponujemy relacją $a(x)$, oznaczającą, że x-tą literą w słowie jest litera **a**. Wtedy na przykład formuła $\exists_x a(x)$ będzie spełniona dla słów, które zawierają co najmniej jedną literę **a**, a formuła $\forall_x \exists_y y \geq x \wedge a(y)$ — dla słów, które kończą się na **a**. W obu przypadkach język złożony ze wszystkich słów, dla których dana formuła jest prawdziwa, jest językiem regularnym — odpowiednie wyrażenia regularne to $A^*\mathbf{a}A^*$ oraz $A^*\mathbf{a}$. Nasuwa się pytanie, czy każdy język regularny daje się zapisać za pomocą formuły logiki pierwszego rzędu, oraz czy każda taka formuła wyznacza jakiś język regularny.

Sformalizujmy trochę pojęcia, z których korzystaliśmy w poprzednim akapicie. Niech A będzie skończonym alfabetem i niech $w = a_0 \dots a_{n-1}$ będzie słowem nad alfabetem A. Słowo w będziemy reprezentować strukturą relacyjną

$$\underline{w} = (\{1, \dots, n\}, \leq(x, y), s(x, y), \{a(x)\}_{\mathbf{a} \in A}),$$

gdzie zbiór $\{1, \dots, n\}$ interpretujemy jako zbiór pozycji w słowie w, s jest relacją następnika, a predykaty $a(x)$ (dla $\mathbf{a} \in A$) wyznaczają pozycje w słowie, na których znajduje się litera **a**.

Przez L_φ będziemy oznaczać język zdefiniowany formułą φ:

$$L_\varphi = \{w \in A^* : \underline{w} \models \varphi\}.$$

Okazuje się, że nie wszystkie języki regularne dają się wyrazić za pomocą formuły logiki pierwszego rzędu. Z poniższego lematu wynika, że nie możemy tak wyrazić np. języka $(\mathbf{aa})^*$.

Lemat 1. *Dla każdej formuły $\varphi \in FO(\leq, s)$ o rozmiarze n zachodzi*

$$\mathbf{a}^{2^n} \models \varphi \iff \mathbf{a}^{2^n+1} \models \varphi.$$

Dowód
Dowód polega na nietrudnej analizie gry Ehrenfeuchta na słowach $\mathbf{a}^{2^n}$ i $\mathbf{a}^{2^n+1}$. Szczegóły pozostawiamy czytelnikowi. □

1.2 Twierdzenie Büchiego

Pokażemy, że odpowiednią logiką do opisywania języków regularnych jest logika MSO (monadic second-order logic), która zezwala na kwantyfikowanie po elementach oraz po zbiorach elementów.

Twierdzenie 2 (Büchi 60, Elgot 61, Trachtenbrot 62)**.** *Dla języka $L \subseteq A^*$ następujące warunki są równoważne:*

1. *L jest regularny*

2. *L jest definiowalny w logice MSO(s).*

Zanim udowodnimy to twierdzenie, stworzymy formułę MSO opisującą język $(\mathbf{aa})^*$, co pozwoli nam dostrzec ogólną metodę konstruowania formuł MSO dla języków regularnych. Słowa o parzystej liczbie liter możemy opisać jako takie słowa, w których możemy zaznaczyć co drugą pozycję tak, by zaznaczyć pierwszą pozycję i nie zaznaczyć ostatniej. Za pomocą formuły logicznej możemy to wyrazić następująco:

$$\exists_X \begin{cases} \forall_x \exists_{y \leq x}\, y \in X & \text{pierwsza pozycja jest zaznaczona} \\ \forall_x \exists_{y \geq x}\, y \notin X & \text{ostatnia pozycja nie jest zaznaczona} \\ \forall_x \forall_y\, s(x,y) \to (x \in X \leftrightarrow y \notin X) & \text{zaznaczona jest co druga pozycja} \end{cases}$$

Przyjrzyjmy się teraz automatowi rozpoznającemu ten język. Automat ten ma dwa stany: akceptujący q_0, będący stanem początkowym i nieakceptujący q_1. Tranzycje opisane literą **a** prowadzą ze stanu q_0 do q_1 oraz z q_1 do q_0. Oznacza to, że przy wczytywaniu słowa należącego do języka, automat po wczytaniu pierwszej pozycji będzie w stanie q_1, po wczytaniu ostatniej pozycji będzie w stanie q_0 (innymi słowy — nie będzie w stanie q_1), oraz będzie w stanie q_0 co dwie pozycje. Zauważmy, że zbiór pozycji, po wczytaniu których automat przejdzie do stanu q_1, jest właśnie zbiorem X spełniającym powyższą formułę. To nie przypadek.

W ogólnym przypadku będziemy tworzyć formułę na podstawie automatu rozpoznającego dany język: będziemy kwantyfikować po zbiorach wyznaczających pozycje, w których automat przechodzi daną tranzycją. W treści formuły wymusimy, by wyznaczone stany odpowiadały akceptującemu biegowi automatu na danym słowie. Poniżej pokażemy to w formalny sposób.

Uwaga. W powyższej formule korzystaliśmy z relacji porządku, ale nie stanowi to nadużycia — porządek jest w MSO definiowalny za pomocą następnika:

$$x \leq y \longleftrightarrow \forall_X (x \in X \wedge (\forall_{u,v}(u \in X \wedge s(u,v)) \to v \in X)) \longrightarrow y \in X$$

(jeśli $x \in X$ i X jest zamknięty ze względu na operację następnika, to $y \in X$).

Dowód („$\Longrightarrow$")
Udowodnimy, że dla każdego automatu skończonego $\mathcal{A} = (Q, Q_I, \delta, A, F)$ istnieje równoważna formuła $\varphi \in MSO(s)$ taka, że $L_{\mathcal{A}} = L_\varphi$. Załóżmy, że tranzycje automatu to $\delta = \{(q_1, a_1, p_1), \ldots, (q_n, a_n, p_n)\}$, gdzie krotka (q_i, a_i, p_i) oznacza „automat będący w stanie q_i po wczytaniu litery a_i przechodzi do stanu p_i". Zauważmy, że każdej pozycji w słowie możemy w naturalny sposób przypisać tranzycję – tę tranzycję, którą przechodzi automat wczytując literę na danej pozycji. Bieg automatu na danym słowie możemy jednoznacznie wyznaczyć określając, na jakich pozycjach automat przechodzi daną tranzycją. Utworzymy teraz formułę, która będzie spełniona dokładnie przez te słowa, dla których istnieje akceptujący bieg automatu:

$$\exists_{X_1} \ldots \exists_{X_n} \begin{cases} \bigwedge_{i \neq j} \forall_x x \in X_i \to x \notin X_j \\ \qquad X_i \text{ są rozłączne} \\ \\ \forall_x \bigvee_i x \in X_i \wedge a_i(x) \\ \qquad X_i \text{ pokrywają słowo, a tranzycje są zgodne z etykietami} \\ \\ \forall_x \forall_y s(x,y) \to (\bigvee_{i,j\,:\,p_i = q_j} x \in X_i \wedge y \in X_j) \\ \qquad \text{kolejne tranzycje są zgodne} \\ \\ \forall_x \exists_{y \leq x} \bigvee_{q_i \in I} y \in X_i \\ \qquad \text{pierwsza tranzycja zaczyna się w stanie inicjalnym} \\ \\ \forall_x \exists_{y \geq x} \bigvee_{p_i \in F} y \in X_i \\ \qquad \text{ostatnia tranzycja kończy się w stanie akceptującym} \end{cases}$$

□

Dowód („$\Longrightarrow$”)
Chcemy pokazać, że dla każdej formuły MSO $\varphi(X_1 \ldots X_n, y_1 \ldots y_n)$ istnieje równoważny automat. Udowodnimy to przez indukcję po budowie φ.

Krok 1. Zacznijmy od uproszczenia formuł MSO – eliminacji zmiennych indywiduowych. Pokażemy przez indukcję ze względu na budowę formuły, że każda formuła MSO $\Psi_{MSO}(x_1, \ldots, x_n, Y_1, \ldots, Y_m)$ posiada równoważną formułę logiki MSO_0 – $\Psi_{MSO_O}(X_1, \ldots, X_n, Y_1, \ldots, Y_m)$, w której posługujemy się tylko zmiennymi drugiego rzędu, oraz dysponujemy predykatem $\subseteq(X,Y)$ oraz $S(X,Y)$ o następującej semantyce:

$$S(X{,}Y)\ {}_{MSO_0} \iff {}_{MSO}\ \exists_{x \in X} \exists_{y \in Y} s(x,y)$$

Skrót notacyjny $\varphi\ {}_{L_1} \iff {}_{L_2} \psi$ będzie oznaczać „formułę φ logiki L_1 przepisujemy na równoważną – prawdziwą w tych samych słowach (a w przypadku formuł otwartych: w tych samych słowach i dla analogicznych wartościowań zmiennych wolnych) – formułę ψ logiki L_2”. Przez „analogiczne” wartościowania rozumiemy takie wartościowania, w których dopuszczamy zmianę wartości zmiennej indywiduowej na singleton zawierający tę wartość. Kwantyfikację po elementach będziemy symulować kwantyfikacją po singletonach, chcemy też w jakiś sposób zastąpić predykaty $s(x,y)$ i $\in X(y)$ nowymi predykatami $S(X,Y)$ i $\subseteq(X,Y)$.

Zdefiniujmy w MSO_0 pomocniczy predykat $sing(X)$, który jest prawdziwy wtw. gdy zbiór X jest jednoelementowy:

$$sing(X) \iff \exists_{Y \subseteq X}(Y \neq X) \wedge \forall_{Z \subseteq X}(Z = X \vee Z = Y)$$

(konieczną i wystarczającą własnością singletonów jest fakt posiadania dokładnie jednego podzbioru właściwego).

Możemy teraz przepisać atomowe formuły MSO w języku MSO_O:

$$\exists_x \varphi_{MSO}(x)\ {}_{MSO} \iff {}_{MSO_0}\ \exists_X sing(X) \wedge \varphi_{MSO_0}(x)$$

$$s(x{,}y)\ {}_{MSO} \iff {}_{MSO_0}\ S(X,Y) \wedge sing(X) \wedge sing(Y)$$

$$x \in Y\ {}_{MSO} \iff {}_{MSO_0}\ X \subseteq Y \wedge sing(X)$$

Zauważmy, że każda formuła MSO_0 daje się zapisać formułą MSO: wystarczy, że zapiszemy $\subseteq(X,Y)$ jako formułę MSO:

$$\subseteq(X,Y)_{MSO_0} \iff {}_{MSO} \forall_{x \in X} Y(x).$$

$S(X,Y)$ już zapisaliśmy w ten sposób. Przejście do MSO_0 nie zwiększa więc siły wyrazu.

Krok 2. Kiedy będziemy indukcyjnie tworzyć automaty dla podformuł naszej formuły, zauważymy, że będziemy musieli tworzyć również automaty dla formuł otwartych, tzn. postaci $\varphi(X_1, \ldots, X_n)$, gdzie $X_1, \ldots, X_n$ są zmiennymi wolnymi. Oprócz słowa będziemy więc musieli dostarczyć na wejściu naszego automatu również wartościowanie tych zmiennych (będących zbiorami pozycji w słowie). W tym celu rozszerzymy alfabet A do alfabetu $A \times \{0,1\}^n$: litera $(\mathbf{a}, c_1, \ldots, c_n)$ na pozycji p będzie oznaczała, że w rozważanym słowie na pozycji p jest litera **a**, oraz że p należy do X_i wtw., gdy $c_i = 1$. Jeśli w naturalny sposób zdefiniujemy operator $w \otimes v$, oznaczający takie właśnie podpisywanie jednego słowa pod drugim, to nowe słowo wejściowe nad rozszerzonym alfabetem będziemy mogli zapisać jako $w \otimes X_1 \otimes \ldots \otimes X_n$.

Przykładowo, dla słowa $w =$ **aababb** oraz zbioru X_1 będącego zbiorem liczb parzystych i zbioru X_2 będącego zbiorem liczb pierwszych uzyskamy następujące słowo nad rozszerzonym alfabetem:

$$\begin{array}{lcllllll} w & = & \mathbf{a} & \mathbf{a} & \mathbf{b} & \mathbf{a} & \mathbf{b} & \mathbf{b} \\ X_1 & = & 0 & 1 & 0 & 1 & 0 & 1 \\ X_2 & = & 0 & 1 & 1 & 0 & 1 & 0 \end{array}$$

Krok 3. Przez indukcję po rozmiarze $\varphi(X_1, \ldots, X_n)$ pokażemy, że język

$$L_\varphi^{X_1,\ldots,X_n} \equiv \{w \otimes X_1 \otimes \ldots \otimes X_n : w \models \varphi(X_1, \ldots, X_n)\} \subseteq (A \times \{0,1\}^n)^*$$

jest regularny – określając automat, który go rozpoznaje.

W łatwy sposób możemy zdefiniować automaty rozpoznające atomowe formuły $X_i \subseteq X_j$, $sing(X_i)$ oraz $S(X_i, X_j)$. Na przykład automat rozpoznający $X_i \subseteq X_j$ sprawdza, czy obecność jedynki w i-tym „wierszu" implikuje obecność jedynki w wierszu j-tym; automaty rozpoznające pozostałe dwa predykaty są równie proste.

Aby wykonać krok indukcyjny, wystarczy że znajdziemy automaty rozpoznające $\varphi \vee \psi$, $\neg\varphi$ oraz $\exists_X \varphi(X)$ dla φ, ψ rozpoznawanym przez automaty skończone – co jest równoważne zamkniętości języków regularnych odpowiednio na sumę, dopełnienie i rzutowanie. Zamkniętość na sumę i dopełnienie jest dobrze znanym faktem, musimy się jednak chwilę zastanowić nad tym, co właściwie oznacza rzutowanie języków regularnych.

Krok 4. Zakładając, że język zdefiniowany formułą $\Psi(X_1, \ldots, X_n)$ jest rozpoznawany przez automat skończony $\mathcal{A}$, chcemy pokazać automat dla języka:

$$\varphi(X_1, \ldots, X_{n-1}) = \exists_{X_n} \Psi(X_1, \ldots, X_n).$$

Szukany automat, działający nad alfabetem $A \times \{0,1\}^{n-1}$, powinien po prostu niedeterministycznie zgadnąć ostatni wiersz zer i jedynek, definiujący zbiór X_n, i na rozszerzonym w ten sposób słowie (już nad alfabetem $A \times \{0,1\}^n$) powinien działać jak automat $\mathcal{A}$. □

Wniosek. Udowodnione właśnie twierdzenie głosi, że dla każdej formuły MSO istnieje równoważny automat. Tworzy również dla każdego automatu równoważną mu formułę MSO postaci

$$\exists_{X_1} \ldots \exists_{X_n} \Psi(X_1, \ldots, X_n),$$

gdzie formuła $\Psi(X_1, \ldots, X_n)$ nie ma kwantyfikatorów po zmiennych drugiego rzędu. Oznacza to, że każda formuła MSO nad sygnaturą słów jest równoważna formule takiej właśnie postaci.

Zadania

Zadanie 1.1. *Zbiór słów $W \subseteq A^*$ nazywamy kodem jeśli każde słowo $w \in A^*$ ma conajwyżej jeden rozkład $w = w_1 \cdots w_n$ dla $w_1, \ldots, w_n$. Dla skończonego zbioru słów W napsiać w logica MSO(s) zdanie φ_W, które ma model wtedy i tylko wtedy gdy W jest kodem.*

Zadanie 1.2. *Pokazać, że dla każdej formuły MSO(s) istnieje równoważna (na słowach skończonych) formuła postaci $\exists X \varphi(X)$, gdzie $\varphi(X)$ jest formułą logiki pierwszego rzędu. Innymi słowy, wystarczy jeden kwantyfikator drugiego rzędu.*

Zadanie 1.3. *Formułę $\varphi(x,y)$ nazwiemy linijką długości d, jeśli jest prawdziwa dokładnie wtedy, gdy odległość między x a y to d. Pokazać, że dla dowolnego n istnieje linijka długości większej niż 2^n, która opisuje się formułą MSO(s) rozmiaru wielomianowego ze względu na n.*

Zadanie 1.4. *Uogólnić powyższe zadanie na funkcje podwójnie, potrójnie i więcej razy wykładnicze. Dokładniejszy opis poniżej. Niech $\exp_k(n)$ będzie k-krotnie wykładniczą funkcją od n, czyli*

$$\exp_0(n) = n \qquad \exp_{k+1}(n) = 2^{\exp_k(n)}.$$

Ustalmy k. Pokazać, że dla dowolnego n istnieje linijka długości większej niż $\exp_k(n)$, która opisuje się formułą MSO(s) rozmiaru wielomianowego ze względu na n. Stałe i wykładnik wielomianu mogą zależeć od k.

Zadanie 1.5. *Korzystając z linijki w poprzednim zadaniu udowodnić, że nie istnieje algorytm rozstrzygający spełnialność dla MSO(s) na słowach skończonych, który by działał w czasie wykładniczym, czy też podwójnie wykładniczym, potrójnie wykładniczym, itd.*

Zadanie 1.6. *Dodajmy do sygnatury relację $x = y + z$. Pokazać, że spełnialność MSO(s,+) na słowach skończonych jest nierozstrzygalna.*

Zadanie 1.7. *Napisać formuły FOL (logiki pierwszego rzędu) definiujące następujące własności:*

a *istnieje dokładnie k elementów w uniwersum,*

b *pomiędzy wierzchołkami u, v istnieje ścieżka długości $\leq k$,*

c *w uniwersum nie ma cyklu długości k,*

Czy można napisać także formuły definiujące następujące własności (na pierwszy rzut oka prostsze wersje podpunktów wyżej)?

a' *uniwersum ma parzyście wiele elementów,*

b' *istnieje ścieżka między wierzchołkami u i v,*

c' *w uniwersum nie istnieje cykl.*

Rozwiązanie na stronie 70.

Rozdział 2

Słowa nieskończone

Spisali: Piotr Szczepański i Krzysztof Kąs

Słowa nieskończone mogą przyjmować różne formy:

1. $A^\omega \quad \omega = \{0,1,2,3\ldots\}$ *np :* *abababdba...*
2. $A^\mathbb{Z}$ *np :* $\ldots abababdba \ldots$
3. $A^\mathbb{R}$

Nas będą interesować jedynie słowa postaci A^ω.

2.1 Twierdzenie Büchiego

Twierdzenie 3 (Büchiego)**.** *Dla języka $L \subseteq A^\omega$ następujące warunki są równoważne:*

1. *L jest definiowalny w logice MSO.*
2. *L jest rozpoznawany przez niedeterministyczny automat z warunkiem Büchiego.*
3. *L można opisać wyrażeniem ω-regularnym.*

Język spełniający trzy warunki z powyższego twierdzenia nazwiemy ω-regularnym.
Dowód powyższego twierdzenia przedstawimy w rozdziale 2.2. Najpierw wyjaśnimy występujące w twierdzeniu pojęcia.

Definicja 1. Wyrażeniem ω-regularnym *nazwiemy skończoną sumę postaci*

$$\bigcup_i L_i K_i^\omega$$

gdzie $L_i, K_i \subseteq A^*$ regularne języki słów skończonych. Dodatkowo przyjmiemy, że $\varepsilon^\omega = \emptyset$.

Ostanie zdanie może budzić lekki niepokój, gdyż $\varepsilon^* = \{\varepsilon\}$. Przyjęcie $\varepsilon^\omega = \emptyset$ ułatwi nam obchodzenie się z wyrażeniami ω-regularnymi.
Poniżej podamy kilka przykładów wyrażeń regularnych wraz z odpowiadającą im formułą MSO:

- A^*aA^ω odpowiada $\exists_x a(x)$
- $(A^*a)^\omega$ odpowiada $\forall_y \exists_{x>y} a(x)$
- $(A^*ba^*b)^\omega$ odpowiada $\forall_x \exists_{y \geq x} b(y) \wedge \exists_{z>y} b(z) \wedge \forall_u y < u < z \Rightarrow a(u)$

Zgodnie z definicją, wyrażeń opatrzonych ω nie można zagnieżdżać, natomiast zapis $A^*aA^\omega + (A^*a)^\omega$ jest jak najbardziej poprawny.

Definicja 2. Warunek Büchiego *dla niedeterministycznych automatów rozpoznających słowa nieskończone brzmi:*

Bieg automatu jest akceptujący, jeśli pewien stan akceptujacy wystapi w nim nieskończenie wiele razy.

Niedeterministyczne automaty z warunkiem Büchiego będziemy oznaczać przez NBA, z angielskiego nondeterministic Büchi automaton.

Można zadać pytanie, czy deterministyczne automaty z warunkiem Büchiego (DBA), są równoważne NBA. Odpowiedź na to pytannie jest negatywna, zachodzi następująca zależność:

$$\text{DBA} \quad \subsetneq \quad \text{NBA}$$

Świadkiem powyższej nierówności jest język opisany wyrażeniem $(a{+}b)^*b^\omega$, inaczej mówiąc: słowa zawierajace skończenie wiele a.

Fakt 4. *Nie istnieje deterministyczny automat z warunkiem Büchiego rozpoznający język* $(a+b)^*b^\omega$.

Dowód
Rozważmy dowolne słowo z tego języka, na przykład b^ω. Pokażemy, że dzięki drobnej modyfikacji zawsze będziemy w stanie oszukać nasz automat. Wczytując słowo, przy założeniu poprawności naszego automatu, na pewnej pozycji i musimy znaleźć się w stanie ackeptującym $q \in F$. Jeśli w naszym słowie podmienimy literę $i+1$ na a, nie zmieni to stanu w pozycji i, z powodu determinizmu. Konstrukcję tę można powtórzyć, zakładając, że litera a nie występuje po pozycji $i+2$. Słowo modyfikujemy w ten sposób, za każdym razem, gdy stwierdzimy, iż znaleźliśmy się w stanie akceptującym. □

2.2 Dowód twierdzenia Büchiego

((3) ⇒ (2)) W poniższej konstrukcji wygodnie korzystać z ϵ-przejść. Łatwo pokazać, że ϵ-przejścia można eliminować z NBA, tak samo jak w automatach na skończonych słowach.

Z powodu niedeterminizmu i ϵ-przejść, języki rozpoznawane przez NBA są zamknięte na sumę. A więc wystarczy wskazać NBA dla języków postaci LK^ω. Niech $\mathcal{A}$ będzie automatem (na słowach skończonych) rozpoznającym L, natomiast $\mathcal{B}$ niech będzie automatem (znów na słowach skończonych) rozpoznającym K.

Skonstruujemy teraz NBA dla języka LK^ω, nazwijmy go $\mathcal{C}$. Stany automatu $\mathcal{C}$ to: stany $\mathcal{A}$, stany $\mathcal{B}$, oraz jeden dodatkowy stan r. Dodatkowy stan r będzie jedynym stanem akceptującym w automacie $\mathcal{C}$. Automat z warunkiem Büchiego dla języka działa w następujący sposób:

1. zaczyna w stanie początkowym automatu $\mathcal{A}$.

2. działa przez pewien czas zgodnie z automatem $\mathcal{A}$

3. w pewnym momencie (wybranym niedeterministycznie, gdy jest w stanie akceptującym automatu $\mathcal{A}$) przechodzi ϵ-przejściem do stanu początkowego automatu $\mathcal{B}$

4. działa przez pewien czas zgodnie w automatem $\mathcal{B}$

5. w pewnym momencie (wybranym niedeterministycznie, gdy jest w stanie akceptującym automatu $\mathcal{B}$) przechodzi ϵ-przejściem do stanu r a następnie ϵ-przejściem do stanu początkowego automatu $\mathcal{B}$.

6. GOTO 4

((2) $\Rightarrow$ (3)) Weźmy dowolny NBA $\mathcal{A}$ o stanach Q, stanie początkowym q_I i stanach akceptujących F. Z twierdzenia Kleenego, dla każdych $p,q \in Q$ istnieje wyrażenie regularne $L_{p,q}$ opisujące te słowa skończone, w których automat $\mathcal{A}$ ma pewien bieg z p do q. Wówczas wyrażenie ω-regularne opisujące słowa akceptowane przez $\mathcal{A}$ to:

$$\bigcup_{q\in F} L_{q_I,q}(L_{q,q})^{\omega}.$$

((2) $\Rightarrow$ (1))

Odpowiednia formuła jest prawie taka sama jak na poprzednim wykładzie. Należy jedynie podmienić ostatni warunek na mówiący, że nieskończenie często występuje tranzycja, której stan docelowy jest akceptujący. Jest to formuła postaci $\forall_x \exists_{y\geq x}\, y \in X$, gdzie X to zbiór pozycji korzystających z tranzycji o docelowym stanie akceptującym.

((1) $\Rightarrow$ (2))

Przeprowadzimy indukcję po strukturze danej formuły.

Należy pokazać, że języki rozpoznawane przez automaty NBA są zamknięte na wszystkie operacje z logiki MSO:

- $\cup$ wynika to z tego, że są to automaty niedeterministyczne.
- $\exists$ (rzutowanie) obsługujemy za pomocą niedeterminizmu, w identyczny sposób jak na poprzednim wykładzie.
- $\cap$ wyraża się przez $\cup$ i $\neg$. Można też zrobić konstrukcję bezpośrednio na automatach, bez korzystania ze skomplikowanej operacji dla dopełnienia.
- $\neg$ (dopełnienie) jest treścią reszty tego wykładu, znajduje się w rozdziale 2.3.

2.3 Dopełnienie dla NBA

W celu zakończenia dowodu twierdzenia Büchiego, wystarczy udowodnić następujący lemat:

Lemat 5. *Niech $L \subseteq A^{\omega}$ będzie językiem rozpoznawanym przez NBA. Wówczas $A^{\omega} - L$ też jest rozpoznawany przez NBA.*

Reszta tego rozdziału to dowód powyższego lematu. Niech $\mathcal{A}$ będzie NBA akceptującym L o zbiorze stanów Q i zbiorze stanów akceptujących F. Chcemy stworzyć automat $\mathcal{B}$ taki, że dla dowolnego $w \in A^{\omega}$:

$$\mathcal{A} \text{ nie akceptuje } w \iff \mathcal{B} \text{ akceptuje } w$$

Równoważnie, $\mathcal{B}$ ma akceptować słowa, w których każdy bieg automatu $\mathcal{A}$ przechodzi skończenie wiele razy przez stany akceptujące z F.

Uwaga. Nie działa następująca naiwna konstrukcja automatu $\mathcal{B}$:

Stanami automatu $\mathcal{B}$ są podzbiory zbioru Q. Stanem automatu $\mathcal{B}$ po przeczytaniu słowa $a_1a_2...a_n$ jest zbiór stanów, jakie $\mathcal{A}$ może mieć po przeczytaniu $a_1a_2...a_n$ (w szczególności stanem początkowym automatu $\mathcal{B}$ jest zbiór stanów początkowych automatu $\mathcal{A}$).

Niech dane będzie słowo nieskończone $w = a_1a_2...$ i niech $P_n \subseteq Q$ będzie stanem automatu $\mathcal{B}$ po przeczytaniu prefiksu $a_1a_2...a_n$. Warunkiem akceptacji przez automat $\mathcal{B}$ słowa w jest:

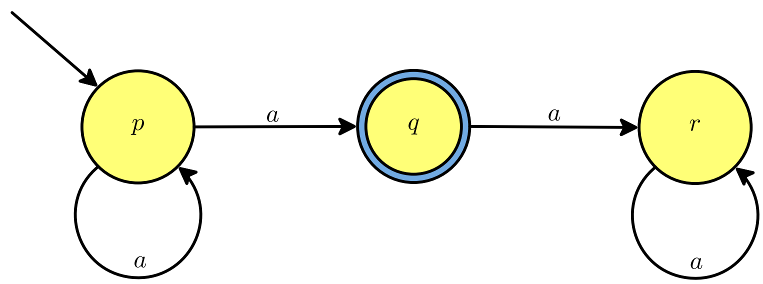
$$\exists_l \forall_{n>l} P_n \cap F = \emptyset$$

(od pewnego momentu już nie ma stanów akceptujących automatu $\mathcal{A}$). Warunek ten nie jest warunkiem typu Büchiego, ale łatwo automat zmienić na NBA: trzeba niedeterministycznie wybierać moment, od którego nie ma stanów akceptujących.

Dlaczego konstrukcja ta nie działa?

Jeśli automat $\mathcal{B}$ akceptuje słowo nieskończone w, to oczywiście $w \notin L$.

Przeciwna implikacja jest jednak fałszywa. Weźmy język $L = \emptyset$ nad alfabetem jednoliterowym $\{a\}$ oraz następujący automat $\mathcal{A}$:



Nie akceptuje on żadnego słowa, bo każdy bieg co najwyżej raz przechodzi przez stan akceptujący. Zatem językiem rozpoznawanym przez $\mathcal{A}$ jest $\emptyset$.

Dla słowa a^ω (jedynego nad tym alfabetem) zachodzi:

$$P_0 = \{p\}$$
$$P_1 = \{p, q\}$$
$$P_2 = \{p, q, r\} = P_3 = P_4 = P_5 = \dots$$

Wobec tego automat $\mathcal{B}$ też nie akceptuje słowa a^ω (a powinien). Wynika to z faktu, że automat $\mathcal{A}$ ma dowolnie długie biegi dochodzące do stanu akceptującego q, ale nie dają się one połączyć w jeden bieg. Potrzebna jest więc inna, subtelniejsza konstrukcja automatu $\mathcal{B}$.

Niech $\mathbb{M}$ oznacza zbiór macierzy A spełniających warunki:

- A jest rozmiaru $|Q| \times |Q|$,
- wiersze i kolumny A są indeksowane przy pomocy stanów automatu $\mathcal{A}$,
- elementy A należą do zbioru $\{\bot; 1; 0\}$.

Każdemu słowu skończonemu $w \in A^*$ przyporządkowujemy pełną informację na jego temat jeśli chodzi o automat $\mathcal{A}$. Ta informacja jest macierzą $M_w \in \mathbb{M}$ zdefiniowaną następująco. W komórce (p,q) wartość macierzy M_w to

$\bot$ jeśli nie istnieje bieg automatu $\mathcal{A}$ na słowie w, zaczynający się w stanie p, który kończy się w stanie q

1 jeśli istnieje bieg automatu $\mathcal{A}$ na słowie w, zaczynający się w stanie p, który kończy się w stanie q i po drodze przechodzi przez stan akceptujący

0 w przeciwnym przypadku, czyli da się przejść z p do q, ale nie da się przechodząc przez stan akceptujący

Aby uzyskać informację na temat konkatenacji $w \cdot v$ (gdzie $w,v \in A^*$) wystarczy znać informację na temat w i v. Inaczej mówiąc, relacja równoważności utożsamiająca słowa o tej samej macierzy jest kongruencją ze względu na konkatenację.

Można to zapisać i uzasadnić definiując mnożenie macierzy w odpowiednim (przemiennym) półpierścieniu. Nośnikiem półpierścienia jest $\{\bot; 1; 0\}$, operacją dodawania jest maksimum według porządku

$$\bot < 0 < 1$$

a mnożenie jest zdefiniowane następująco:

$$\bot \cdot \bot = 1 \cdot \bot = \bot \cdot 1 = 0 \cdot \bot = \bot \cdot 0 = \bot$$
$$1 \cdot 1 = 1 \cdot 0 = 0 \cdot 1 = 1$$
$$0 \cdot 0 = 0$$

Mnożenie macierzy jest zdefiniowane następująco:

$$(M\cdot N)(p,q) = \max_{r\in Q}(M(p,r))\cdot(N(r,q)).$$

Po wprowadzeniu tej terminologii możemy napisać, że:

$$M_{w\cdot v} = M_w\cdot M_v$$

Inaczej mówiąc, przyporządkowanie $w \mapsto M_w$ jest homormofizmem monoidów (z lewej wolnego A^*, a z prawej $\mathbb{M}$). Aby to udowodnić, wystarczy zauważyć, że:

- $M_{w\cdot v}(p,q) = \perp \iff$ dla każdego stanu $r \in Q$ jedno z dwóch przejść jest niemożliwe: (po słowie w od p do r) oraz (po słowie v od r do q) $\iff$ dla każdego stanu $r \in Q$ zachodzi $M_w(p,r) = \perp$ lub $M_v(r,q) = \perp$

- $M_{w\cdot v}(p,q) = 1 \iff$ istnieje stan $r \in Q$ taki, że są możliwe przejścia (po słowie w od p do r) oraz (po słowie v od r do q) a ponadto jedno z nich przechodzi przez stan akceptujący $\iff$ istnieje stan $r \in Q$ taki, że zbiór $\{M_w(p,r), M_v(r,q)\}$ jest równy $\{1\}$ lub $\{0,1\}$

Lemat 6. *Dla dowolnego automatu $\mathcal{A}$ i macierzy $M \in \mathbb{M}$ z odpowiadającego $\mathcal{A}$ zbioru $\mathbb{M}$, regularny jest język*

$$L_M = \{w \in A^* : M_w = M\}.$$

Dowód: Automat czyta słowo i oblicza macierz odpowiadającą już przeczytanemu prefiksowi. Ten automat jest zdefiniowany następująco:

- stanami są wszystkie możliwe macierze z $\mathbb{M}$
- stanem początkowym jest macierz M_ϵ
- ze stanu N po przeczytaniu litery a automat przechodzi do stanu $N\cdot M_a$ (macierz M_a jest macierzą dla słowa jednoliterowego a)

Jeśli stanem akceptującym uczynimy M, automat akceptuje dokładnie słowa ze zbioru L_M. □

Teraz sformułujemy kluczowy lemat.

Lemat 7 (Kluczowy)**.** *Niech $w \in A^\omega$. Wówczas istnieją*

- *macierze M, N takie, że $M\cdot N = M$ i $N\cdot N = N$,*
- *podział $w = w_0w_1w_2w_3\ldots$, gdzie $w_i \in A^+$,*

takie, że

- $M_{w_0} = M$,
- $M_{w_i} = N$ *dla* $i = 1, 2, 3, \ldots$.

A więc:

- *dla dowolnego i słowu $w_0w_1w_2\ldots w_i$ odpowiada macierz M,*
- *dla dowolnych $i \le j$ słowu $w_iw_{i+1}w_{i+2}\ldots w_j$ odpowiada macierz N.*

Zanim przystąpimy do dowodu, przypomnimy twierdzenie Ramseya, które się przyda w dowodzie lematu.

Twierdzenie 8 (Ramseya)**.** *Mamy dany pełny graf nieskierowany, którego wierzchołkami są liczby naturalne. Ponadto każda krawędź (i,j) (ponieważ krawędzie są nieskierowane, to zakładamy, że $i < j$) jest pokolorowana kolorem $\alpha(i,j) \in K$, gdzie K jest skończonym zbiorem kolorów. Wówczas istnieje zbiór nieskończony $X \subseteq \mathbb{N}$ oraz kolor $k \in K$ takie, że*

$$\forall_{i,j \in X}\ \alpha(i,j) = k.$$

Dowód (kluczowego lematu)
Niech $w = a_1 a_2 a_3 \ldots$. Zdefiniujmy $\alpha : [\mathbb{N}]^2 \to \mathbb{M}$ następująco:

$$\alpha(\{i < j\}) = M_{a_i \ldots a_{j-1}} \qquad \text{dla } i < j$$

Na mocy tw. Ramseya istnieją: nieskończony zbiór $I \subseteq \mathbb{N}$ oraz macierz $N \in \mathbb{M}$ takie, że:

$$\forall_{i,j \in I,\ i<j} \alpha(i,j) = N$$

Niech $I = \{i_1, i_2, i_3, \ldots\}$. Zdefiniujmy skończone słowa

$$v_0 = a_1 \ldots a_{i_1 - 1} \quad v_1 = a_{i_1} \ldots a_{i_2 - 1} \quad v_2 = a_{i_2} \ldots a_{i_3 - 1} \quad \cdots$$

Niech N_i macierz opisująca słowo v_i. Wiemy, że wszystkie macierze N_i są równe dla $i \geq 1$. Wybierzmy sobie

$$M = N_0 \cdot N_1 \qquad N = N_1 = N_2 = \cdots$$

Macierze te odpowiadają podziałowi

$$w = w_0 w_1 w_2 \cdots \qquad \text{dla} \quad w_0 = v_0 v_1 \quad w_1 = v_2 \quad w_2 = v_3 \quad \cdots$$

Aby zakończyć dowód, trzeba jescze pokazać warunki na mnożenie macierzy z tezy lematu:

$$N \cdot N = N_1 \cdot N_2 = N \qquad \text{z tezy tw. Ramseya}$$

$$M \cdot N = N_0 \cdot N \cdot N = N_0 \cdot N = M$$

□

Powiemy, że para macierzy N,M spełnia warunek $(\star)$ jeśli istnieje $q \in Q$ takie, że:

- $M(q_I,q) \neq \perp$ dla stanu inicjalnego q_I automatu $\mathcal{A}$
- $N(q,q) = 1$

Lemat 9 (Charakteryzujący)**.** *Ustalmy niedeterministyczny automat z warunkiem Büchiego $\mathcal{A}$ oraz słowo $w \in A^\omega$. Niech macierze M, N i podział $w = w_0 w_1 w_2 \ldots$ będą takie, jak w dowodzie kluczowego lematu. Wówczas słowo w jest akceptowane przez automat $\mathcal{A}$ wtedy i tylko wtedy gdy macierze M, N dodatkowo spełniają warunek $(\star)$.*

Dowód
Dowód implikacji $(\Leftarrow)$ jest oczywisty — warunek $(\star)$ wprost przekłada się na istnienie biegu akceptującego. Przejdźmy więc do implikacji $(\Rightarrow)$.

Rozważmy bieg akceptujący automatu $\mathcal{A}$ na słowie w. Niech q_i oznacza stan automatu w tym biegu po przeczytaniu prefiksu $w_0 w_1 \ldots w_i$. Ponieważ macierz opisująca ten prefiks to M (z założenia $MN = M$), wnioskujemy, że $M(q_I,q_i) \neq \perp$ dla każdego $i \geq 1$. Wówczas istnieje taki stan q, że $q = q_i$ dla nieskończenie wielu i (niech I będzie zbiorem takich właśnie i). W szczególności $M(q_I,q) \neq \perp$, więc spełniona jest pierwsza część warunku $(\star)$. Ponieważ jest to też bieg akceptujący, to nieskończenie często pojawia się w tym automacie pewien stan akceptujący f. Zatem można wybrać taki nieskończony zbiór $J \subseteq I$, $J = \{j_0, j_1, j_2, \ldots\}$, że dla każdego k stan akceptujący f pojawia się podczas czytania przez automat podsłowa $w_{j_k+1} \ldots w_{j_{k+1}}$. Innymi słowy, macierz słowa $w_{j_k+1} \ldots w_{j_{k+1}}$ ma 1 w komórce (q,q). Ponieważ ta macierz to N, mamy $N(q,q) = 1$, więc spełniona jest druga część warunku $(\star)$. □

Wniosek 10. *Dla słowa $w \in A^\omega$ następujące warunki są równoważne:*

1. *w nie jest akceptowane przez $\mathcal{A}$*

2. *istnieją $M, N \in \mathbb{M}$ oraz $w = w_0 w_1 w_2 \dots$ jak w kluczowym lemacie takie, że M, N nie spełniają ($\star$), inaczej mówiąc słowo należy do języka opisanego wyrażeniem regularnym*

$$\bigcup_{\substack{M,N \text{ nie spełniają warunku } (\star) \\ M \cdot N = M \\ N \cdot N = N}} L_M (L_N)^\omega$$

To kończy dowód tw. Büchiego.

Zadania

Zadanie 2.1. *Pokazać, że każdy niedeterministyczny automat z warunkiem Mullera jest równoważny pewnemu NBA.* *Rozwiązanie na stronie 72.*

Zadanie 2.2. *Pokazać, że języki rozpoznawane przez deterministyczne automaty z warunkiem Büchiego są zamknięte na przecięcie.* *Rozwiązanie na stronie 73.*

Zadanie 2.3. *Pokazać, że następujący język nie jest ω-regularny:*

$$\{a^{n_1}ba^{n_2}b\cdots : \lim_{i\to\infty} n_i = \infty\}.$$

Rozwiązanie na stronie 74.

Zadanie 2.4. *Słowo ostatecznie okresowe to słowo postaci wv^ω dla $w,v \in A^+$. Pokazać, że jeśli dwa języki ω-regularne zawierają te same słowa ostatecznie okresowe, to muszą być równe.* *Rozwiązanie na stronie 75.*

Zadanie 2.5. *Wskazać język ω-regularny, który nie jest różnicą dwóch języków rozpoznawanych przez deterministyczne automaty z warunkiem Büchiego.* *Rozwiązanie na stronie 77.*

Zadanie 2.6. *Zaprojektować algorytm, który sprawdza czy język rozpoznawany przez dany NBA jest przeliczalny.* *Rozwiązanie na stronie 78.*

Zadanie 2.7. *Rozważmy trzy relacje à la Myhill Nerode, które są zadane przez język słów nieskończonych $L \subseteq A^\omega$, ale porównują słowa skończone $w,w' \in A^*$.*

$$w \sim_L^1 w' \qquad \text{jeśli} \qquad uwv \in L \iff uw'v \in L \text{ dla każdych } u \in A^*, v \in A^\omega$$

$$w \sim_L^2 w' \qquad \text{jeśli} \qquad \wedge \begin{cases} uwv \in L \iff uw'v \in L & \text{dla każdych } u \in A^*, v \in A^\omega \\ u(wv)^\omega \in L \iff u(w'v)^\omega \in L & \text{dla każdych } u,v \in A^* \end{cases}$$

$$w \sim_L^3 w' \qquad \text{jeśli} \qquad u \in L \iff u' \in L \quad \begin{array}{l} \text{gdzie } u' \in A^\omega \text{ otrzymano z } u \in A^\omega \\ \text{przez zamienienie (skończonej lub nie)} \\ \text{ilości rozłącznych wystąpień } w \text{ na } w'. \end{array}$$

Niech $i = 1,2$. Wskazać język L, który ma skończenie wiele klas abstrakcji w relacji w relacji $\sim_L^i$, ale nieskończenie wiele klas abstrakcji w relacji $\sim_L^{i+1}$. Wskazać język L, który nie jest ω-regularny, ale dla którego $\sim_L^3$ ma skończenie wiele klas abstrakcji.

Rozdział 3

Interpretacje

Spisali: Michał Skrzypczak

W tym rozdziale rozpatrywać będziemy logiki FO i MSO, w strukturach relacyjnych. Kluczowym pojęciem będzie *rozstrzygalność teorii*.

Definicja 3. *Powiemy, że teoria ustalonej logiki, przy ustalonej signaturze Σ, w modelu M jest* **rozstrzygalna**, *jeśli istnieje algorytm wczytujący dowolną formułę φ i odpowiadający, czy $M \models \varphi$.*

W zależności od logiki, sygnatury i modelu, problem spełnialności może być rozstrzygalny lub nie. Na przykład dla logiki składającej się z jednej formuły $\perp$, nie spełnionej w żadnym modelu, problem ten jest zawsze rozstrzygalny. Z drugiej strony problem spełnialności formuł logiki pierwszego rzędu w liczbach naturalnych z dodawaniem i mnożeniem jest nierozstrzygalny (Gödel).

Celem będzie umiejętność przenoszenia rozstrzygalności problemu spełnialności z jednego modelu, na inny. Na przykład wiemy, że liczby naturalne z następnikiem mają rozstrzygalną teorię MSO. Chcielibyśmy z tego wywnioskować, że również liczby całkowite mają rozstrzygalną teorię MSO. Metodą dowodzenia takich twierdzeń będą *interpretacje*.

Definicja 4. *Powiemy, że struktura relacyjna M_1 nad sygnaturą Σ_1* **interpretuje** *się w strukturze M_2 nad sygnaturą Σ_2, jeśli istnieją:*

- *różnowartościowe zanurzenie $i : |M_1| \to |M_2|$, gdzie $|M_j|$ oznacza nośnik odpowiedniej struktury,*
- *formuła logiczna $p(x)$ nad sygnaturą Σ_2, spełniona dla $x \in |M_2|$ wtedy i tylko wtedy, gdy x leży w obrazie i,*
- *dla każdego symbolu relacyjnego $R \in \Sigma_1$ formuła $\varphi_R(x_1, \ldots, x_n)$ nad sygnaturą Σ_2, taka że dla każdych $x_1, \ldots, x_n \in |M_1|$, mamy $R(x_1, \ldots, x_n)$ wtedy i tylko wtedy, gdy $M_2 \models \varphi_R(i(x_1), \ldots, i(x_n))$.*

Nie wymagamy żadnych własności od włożenia i, może to być dowolna, teoriomnogościowa funkcja. Kluczem powyższej definicji jest możliwość mapowania dowolnej formuły φ nad sygnaturą M_1, na formułę φ' nad sygnaturą Σ_2, w ten sposób, by $M_1 \models \varphi$ wtw. $M_2 \models \varphi'$. Mapowanie to odbywa się w ten sposób, że w miejsce wystąpienia dowolnego symbolu relacyjnego R wstawiamy odpowiednią formułę φ_R, a kwantyfikatory wzbogacamy o wymaganie, by kwantyfikowana zmienna x spełniała warunek $p(x)$. Dzięki temu, otrzymujemy następujące twierdzenie.

Twierdzenie 11. *Jeśli struktura M_1 interpretuje się w strukturze M_2 i teoria M_2 jest rozstrzygalna, wtedy rozstrzygalna jest również teoria M_1.*

Dowód

Można tu bezpośrednio skonstruować algorytm. Zakładamy, że dana jest formuła φ. Obliczamy formułę φ', zgodnie z podaną konstrukcją. Sprawdzamy, czy φ' jest spełnione w M_2. Odpowiedź jest jednocześnie odpowiedzią na pytanie, czy φ było spełnione w M_1. $\square$

Klasycznym przykładem zastosowania powyższej metody, jest dowód, że teoria MSO jest rozstrzygalna w liczbach całkowitych z następnikiem. Możemy mianowicie interpretować liczby całkowite w liczbach naturalnych, z włożeniem

$$i(k) = \begin{cases} 0 & \text{dla } k = 0 \\ 2k & \text{dla } k > 0 \\ -2k - 1 & \text{dla } k < 0 \end{cases}$$

Odpowiednie formuły dla relacji następnika i zera można podać jako proste ćwiczenie.

Zdefiniowane powyżej interpretacje nazywane są też interpretacjami typu *element-element*. Znajdują one zastosowanie dla wszystkich logik nad sygnaturami relacyjnymi. Możemy jednak ograniczyć stosowalność do szczególnych logik i rozszerzyć pojęcie interpretacji.

Definicja 5. *Powiemy, że model M_1 interpretuje się w M_2 typu element-zbiór, jeśli istnieje włożenie $i : |M_1| \to 2^{|M_2|}$ oraz formuły monadyczne $p(X)$ i φ_R analogicznie jak w zwykłych interpretacjach.*

Dzięki powyższej definicji możemy przenosić rozstrzygalność teorii MSO w M_2 na rozstrzygalność FO w M_1. Ciekawy przykład wykorzystania tej metody, to interpretacja liczb naturalnych z dodawaniem w liczbach naturalnych z następnikiem, metodą element-zbiór. Główną ideą jest kodowanie binarne, zaczynając od końca. Implikuje to rozstrzygalność logiki pierwszego rzędu dla arytmetyki Presburgera.

Rozdział 4

Determinizacja

Spisali: Michał Skrzypczak

W przypadku automatów dla słów skończonych, powszechnie znana jest konstrukcja automatu deterministycznego, równoważnego danemu automatowi niedeterministycznemu $\mathcal{A}$. Podstawową ideą jest, by tworzony automat $\mathcal{A}'$ miał za stany zbiory stanów automatu $\mathcal{A}$, a przejścia mapowały zbiory stanów w zbiory stanów. Daje to wykładniczy wzrost rozmiaru automatu i nietrudno udowodnić, że nie da się tego w ogólnym przypadku poprawić. Automat ten nazywa się automatem potęgowym.

W tym rozdziale postaramy się przeprowadzić analogiczną konstrukcję w przypadku automatów dla słów nieskończonych. W tym celu potrzebny będzie nowy, nieco inny niż do tej pory warunek akceptacji.

Definicja 6. *Deterministyczny automat z warunkiem Mullera, to standardowy deterministyczny automat dla słów nieskończonych o zbiorze stanów Q, z wyróżnioną rodziną zbiorów $F_1,F_2,\ldots,F_j \subseteq Q$. Powiemy, że automat taki akceptuje nieskończone słowo w, jeśli zbiór stanów występujących nieskończenie często w biegu na w jest jednym ze zbiorów $F_1,F_2,\ldots,F_j$.*

Alternatywnie (równoważnie) można mówić, że zamiast rodziny zbiorów F_i, automat taki ma formułę boolowską (czyli formułę z operacjami $\wedge$, $\vee$, $\neg$) w której zdaniami atomowymi są zdania postaci *„bieg przechodzi nieskończenie wiele razy przez stan q”*. Przy takiej definicji, mówimy, że automat akceptuje, jeśli cała formuła jest prawdziwa.

Celem wykładu będzie wykazanie następującego twierdzenia.

Twierdzenie 12. *Dla każdego niedeterministycznego automatu z warunkiem Büchiego istnieje równoważny deterministyczny automat z warunkiem Mullera.*

4.1 Podejście bezpośrednie

Zauważmy najpierw, że naiwne przeniesienie konstrukcji dla słów skończonych nic nie daje. W naiwnej konstrukcji stosujemy automat potęgowy tak jak w słowach skończonych, a stanami akceptującymi czynimy te zbiory, które zawierają przynajmniej jeden stan akceptujący.

Rozpatrzmy automat $\mathcal{A}$ nad alfabetem jednoliterowym $\{a\}$ o stanach p,q,r i przejściach po literce a:

$$p \xrightarrow{a} p \qquad p \xrightarrow{a} q \qquad q \xrightarrow{a} r \qquad r \xrightarrow{a} r.$$

Stan q jest jedynym stanem akceptującym. Rozpatrzmy warunek akceptacji Büchiego (chociaż to nie ma wielkiego znaczenia jaki warunek weźmiemy). Nad aflabetem jednoliterowym $\{a\}$ istnieje tylko jedno słowo a^ω. W każdym biegu na a^ω nasz automat przechodzi co najwyżej raz przez stan akceptujący. W związku z tym nie akceptuje on tego słowa. Jeśli natomiast skonstruujemy automat potęgowy, o stanach $2^{\{p,q,r\}}$, to przejścia wyglądać będą następująco:

$$\{p\} \xrightarrow{a} \{p,q\} \qquad \{p,q\} \xrightarrow{a} \{p,q,r\} \qquad \{p,q,r\} \xrightarrow{a} \{p,q,r\}.$$

Czyli już po wczytaniu trzeciej literki a automat na zawsze pozostanie w stanie $\{p,q,r\}$, z czego nie daje się wywnioskować, czy pierwotny automat akceptuje, czy nie.

Problem z automatem potęgowym polega na tym, że pojawienie się stanów akceptujących w nieskończenie wielu zbiorach oznacza tylko tyle, że istnieją biegi oryginalnego automatu niedeterministycznego, które mają dowolnie daleko stan akceptujący. Niekoniecznie można z tych biegów stworzyć jeden bieg mający stany akceptujące nieskończenie często, czy choćby dwa razy, jak w powyższym przykładzie. Widzimy więc, że potrzebna będzie subtelniejsza konstrukcja, zdolna do wychwycenia sytuacji takich jak powyższa.

Subtelniejsza konstrukcja przedstawiona w tym wykładzie nazywa się konstrukcją Safry i jest opisana poniżej.

4.2 Drzewa Safry

Podstawową ideą omawianej konstrukcji jest, by tak wzbogacić strukturę informacji pamiętanych w stanach automatu, by dało się z nich wywnioskować kiedy miało miejsce przejście przez stan akceptujący. Okazuje się, że właściwą konstrukcją są *drzewa Safry.*

Definicja 7. *Drzewo Safry dla zbioru stanów Q, to skończone drzewo T złożone z podzbiorów Q. Wierzchołki będziemy oznaczać przez x,y,z, a ich etykiety przez $T(x)$,$T(y)$,$T(z) \subseteq Q$. Dodatkowo każdy wierzchołek ma przypisany identyfikator ze zbioru $\{0,1,2,\ldots,2|Q|-1\}$. Niektóre z wierzchołków T są oznaczone jako świeże. Jak to w drzewie używamy następujących określeń: syn, ojciec, brat, potomek (domknięcie przechodnie syna). Dla relacji potomka używamy znaku $x < y$, gdzie y jest potomkiem x. Oprócz tego zakładamy, że w obrębie synów każdego wierzchołka wprowadzony jest pewien porządek liniowy (czyli są synowie starsi i młodsi). Dodatkowo wymagamy następujących warunków:*

1. *Suma etykiet synów jest właściwym podzbiorem etykiety ojca.*
2. *Zbiory stanów w etykietach braci są rozłączne.*
3. *Każdy wierzchołek ma niepustą etykietę.*

Nie ma żadnych warunków na to które wierzchołki drzewa są świeże, ani jak przydzielone są identyfikatory.

Pisząc o drzewie Safry T, myślimy o wszystkich informacjach naraz: zbiór wierzchołków, struktura drzewa, porządek liniowy na braciach, etykiety wierzchołków, identyfikatory wierzchołków, oraz zbiór wierzchołków świeżych. Łatwo sprawdzić, że zbiór wszystkich możliwych drzew Safry dla określonego zbioru stanów Q jest skończony. Można wykazać, że jego rozmiar wynosi asymptotycznie $2^{|Q|\log(|Q|)}$.

4.3 Symulacja

Opiszemy teraz jak można symulować za pomocą drzew Safry bieg danego $\mathcal{A}$ na słowach nieskończonych. Rozpoczniemy z drzewem Safry o jednym węźle, będącym singletonem stanu początkowego w $\mathcal{A}$. Węzeł ten nie będzie świeży, a jego identyfikator będzie to np. 0. Oznaczamy takie drzewo T_0.

Teraz opiszemy jak zmienia się dane dowolne drzewo Safry T pod wpływem wczytania nowej litery a. Będzie się to odbywało w kilku krokach:

1. Stosujemy automat potęgowy do każdej etykiety drzewa. Inaczej mówiąc, dla każdego wierzchołka x drzewa T zmieniamy etykietę z $T(x)$ na

$$\{p : \text{ w automacie } \mathcal{A} \text{ jest przejście } q \xrightarrow{a} p \text{ dla pewnego } q \in T(x)\}.$$

   Nie zmieniamy identyfikatorów. Warunki 1, 2 i 3 mogą być potencjalnie niespełnione.

2. Dla każdego wierzchołka x drzewa T tworzymy nowego syna y, składającego się ze wszystkich stanów akceptujących zawartych w tym momencie w x (czyli już po zastosowaniu automatu potęgowego). Nowy wierzchołek porządkujemy jako najmłodszego. Nadajemy mu jakiś nowy, wolny identyfikator. Wciąż warunki 1, 2 i 3 mogą być potencjalnie niespełnione.

3. Dla każdego wierzchołka x i każdego stanu q, jeśli q występuje też w jakimś starszym bracie wierzchołka x, to usuwamy q z x i potomków x. Kolejność wyboru x nie ma znaczenia. Teraz warunek 2 jest już spełniony, natomiast warunki 1 i 3 wciąż mogą być niespełnione.

4. Usuwamy wszystkie wierzchołki o pustych zbiorach stanów. Jedyny warunek jaki może teraz być niespełniony, to 1.

5. Wycieramy z drzewa oznaczenie świeżości, tak by żaden węzeł nie był świeży.

6. Dla każdego wierzchołka x jeśli suma etykiet dzieci x daje etykietę x, to kasujemy wszystkie dzieci x wraz z potomkami i oznaczamy x jako świeży. W tym momencie wszystkie warunki jakie ma spełniać drzewo Safry są spełnione.

W ten sposób zdefiniowaliśmy nowe drzewo Safry S, powstałe z T. Powyższą konstrukcję modyfikacji T przez literkę a oznaczamy $T \xrightarrow{a} S$. Rozszerzamy tę notację na skończone słowa, do $T \xrightarrow{w} S$.

Poniżej udowodnimy kilka dodatkowych warunków jakie powyższa konstrukcja gwarantuje. Weźmy dowolne $w \in \Sigma^*$ i weźmy T takie by $T_0 \xrightarrow{w} T$.

1. T jest niepuste (ma korzeń) wtw. istnieje bieg automatu $\mathcal{A}$ na słowie w.

2. Jeśli T jest niepuste, to etykieta korzenia składa się z dokładnie tych stanów $q \in Q$, które są osiągalne z q_0 po przejściu przez słowo w.

3. Jeśli przy przejściu $T \xrightarrow{a} S$ wierzchołek $x \in T$ zostaje skasowany, to w S nie ma wierzchołka o takim identyfikatorze jak x.

4. Gdy wierzchołek powstaje to zawsze zawiera same stany akceptujące.

5. Wierzchołek może zostać skasowany na dwa sposoby, albo gdy staje się pusty, albo gdy jego ojciec staje się świeży.

4.4 Automat

Pozostaje, w oparciu o drzewa Safry, zdefiniować deterministyczny automat Mullera równoważny danemu niedeterministycznemu automatowi Büchiego. Przyjmijmy za jego stany wszystkie drzewa Safry, stan początkowy niech będzie równy T_0, a przejścia będą zgodne z opisanymi powyżej. Poniższy lemat pokazuje jaki warunek akceptacji należy rozpatrzyć w tworzonym automacie.

Lemat 13. *Weźmy słowo nieskończone $w = a_1 a_2 \cdots$. Rozważmy drzewa Safry*

$$T_0 \xrightarrow{a_1} T_1 \xrightarrow{a_2} T_2 \xrightarrow{a_3} T_3 \xrightarrow{a_4} \cdots$$

Równoważne są warunki:

a) *istnieje bieg akceptujący $\mathcal{A}$ na w,*

b) *istnieje identyfikator wierzchołka $j \in \{0,1,\ldots,2|Q|-1\}$, taki że od pewnego momentu wierzchołek o takim identyfikatorze zawsze jest obecny w T_i i nieskończenie wiele razy wierzchołek o tym identyfikatorze jest świeży.*

Łatwo zobaczyć, że warunek w punkcie b) jest typu Mullera, więc jeśli mamy powyższy lemat, to dowód jest zakończony, gdyż opisany powyżej automat Safry rozpoznaje dokłądnie te same słowa co pierwotny automat $\mathcal{A}$.

Dowód z a) do b). Rozważmy bieg akceptujący

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} q_3 \xrightarrow{a_4} \cdots$$

Oznaczmy przez x_i najgłębszy wierzchołek drzewa T_i, w którego etykiecie znajduje się stan q_i. Z poczynionych wcześniej obserwacji wynika, że taki wierzchołek zawsze istnieje. (Jest też jednoznaczny, z rozłącznośći etykiet braci).

Oznaczmy korzeń drzewa T_i przez y_i. Oczywiście $q_i \in T_i(y_i)$ dla każdego i. Skoro korzenie nie są nigdy kasowane, wszystkie wierzchołki y_i mają ten sam identyfikator. Jeśli nieskończenie wiele razy y_i jest świeży, to koniec, bo dla identyfikatora korzenia zachodzi b). Załóżmy przeciwnie. W takim razie od pewnego momentu, nazwijmy ten moment i_0, wierzchołek y_i nie jest nigdy odświeżany. Jednocześnie nieskończenie często q_i jest stanem akceptującym. Znaczy to, że nieskończenie często po momencie i_0 wierzchołek y_i znajduje się w nowo stworzonym wierzchołku, czyli głębiej niż w korzeniu. Po momencie i_0 wierzchołek x_i nie może wrócić do korzenia, a więc od pewnego momentu wierzchołek x_i znajduje się poza korzeniem. Ponieważ wśród synów korzenia nie nieskończonego ciągu starzejącego się, istnieje syn korzenia, nazwijmy go z_i, nie kasowany od pewnego momentu, w którym od pewnego momentu zawsze zachodzi $x_i \geq z_i$.

Możemy teraz rozumowanie z powyższego akapitu powtórzyć dla z_i. Albo jest on nieskończenie często świeży, albo ma syna. I tak dalej.... Ale drzewa T_i mają ograniczoną wysokość, więc w pewnym momencie musimy przerwać postępowanie, znalazłszy wierzchołek który od pewnego momentu zawsze istnieje i jest nieskończenie wiele razy świeży. □

Aby wykazać implikację z b) do a) najpierw udowodnimy lemat.

Lemat 14. *Weźmy dowolne drzewo Safry T. Załóżmy, że przechodzimy po słowie $w \neq \epsilon$ do drzewa S. Załóżmy, że x jest świeży w T,S i nie jest kasowany w trakcie. Wtedy dla każdego stanu $p \in S(x)$ istnieje stan $q \in T(x)$ i bieg po w z q do p przechodzący przez stan akceptujący.*

Dowód. Patrzymy na stan $p \in x$ w S. Skoro x jest świeży w S, to istnieje stan p' który w poprzednim drzewie należał do pewnego syna x oznaczonego x' i ostatnia literka w może przenieść go w p. Patrzymy na historię x', w szczególności na ciąg stanów prowadzących do p'. Dopóki idąc wstecz x' istnieje, wszystko dobrze. W pewnym momencie x' powstał, bo w T x nie miał dzieci. To był moment w którym wszystkie stany należące do x' były akceptujące. W szczególności pewien poprzednik p'. Cofamy się dalej do T uzyskując szukany wierzchołek q wraz z biegiem od q przez stan akceptujący do p' i wreszcie do p, po słowie w. □

Możemy teraz zakończyć dowód twierdzenia.

Dowód z b) do a). Istnieje wierzchołek x który od pewnego momentu nie jest nigdy skasowany i jest świeży w momentach i_1,i_2,....

Zdefiniujmy zbiór H takich skończonych ciągów stanów $(p_1,p_2,\ldots,p_n)$, że istnieje bieg automatu $\mathcal{A}$ na słowie w, który:

- dla każdego $1 \leq k \leq n$ w momencie i_k jest w stanie p_k,
- dla każdego $1 < k \leq n$ pomiędzy momentami i_{k-1} i i_k przechodzi przez jakiś stan akceptujący.

Jak widać z definicji H jest zamknięty ze względu na prefiksy, więc tworzy drzewo. Dodatkowo drzewo to ma skończone rozgałęzienie, gdyż występują w nim tylko znaki ze skończonego zbioru Q.

Korzystając z lematu, dla każdego $n > 0$ istnieje stan q osiągalny w momencie i_n (ściślej należący do x w drzewie T_{i_n}) oraz taki bieg z q_0 do q, który przechodzi przez stany akceptujące pomiędzy momentami i_{k-1} i i_k dla $1 < k \leq n$. Więc drzewo H zawiera dowolnie długie ścieżki. Więc posiada pewną nieskończoną gałąź. Gałąź ta odpowiada nieskończonemu biegowi $\mathcal{A}$ na słowie w, który przechodzi nieskończenie wiele razy przez stany akceptujące. □

4.5 Twierdzenie odwrotne

Ma miejsce również twierdzenie odwrotne do wykazanej powyżej determinizacji.

Twierdzenie 15. *Dla każdego deterministycznego automatu Mullera $\mathcal{A}$ istnieje równoważny niedeterministyczny automat Büchiego.*

Dowód. Z definicji $\mathcal{A}$ jest boolowską kombinacją φ niedeterministycznych automatów Büchiego. A jak sprawdziliśmy automaty te są zamknięte ze względu na operacje logiczne. Więc można skonstruować automat równoważny całej formule φ. Jest on równoważny $\mathcal{A}$. □

W tym rozumowaniu korzystamy istotnie z faktu, że niedeterministyczne automaty Büchiego są zamknięte na dopełnienia. Można stworzyć odpowiednią konstrukcję bardziej bezpośrednio, nie korzystając z tego faktu. Oznaczmy przez $\mathcal{A}$ dany deterministyczny automat Mullera. Poniżej znajduje się opis konstrukcji niedeterministycznego automatu Büchiego $\mathcal{B}$, który akceptuje te same słowa.

$\mathcal{B}$ zgaduje który ze zbiorów F_i będzie realizować. Formalnie rzecz biorąc, język rozpoznawany przez $\mathcal{B}$ jest sumą języków automatów $\mathcal{B}_i$ odpowiadających zbiorom F_i. A wiemy, że suma jest łatwa dla automatów nideterministycznych. Załóżmy, że $F_i = \{f_0, f_1, \ldots, f_{k-1}\}$. Bieg składa się z dwóch faz:

- Najpierw $\mathcal{B}_i$ po prostu symuluje bieg $\mathcal{A}$. W tej fazie $\mathcal{B}_i$ nie ma żadnych stanów akceptujących. Następnie w pewnym momencie automat zgaduje, że stany spoza F_i nie będą się już pojawiać i przechodzi do drugiej fazy, która odbywa się na kopii $\mathcal{A}$ składającej się tylko ze stanów F_i.

- W tej fazie automat stale pamięta na który ze stanów $f_j \in F_i$ czeka. W momencie przejścia przez szukany stan f_i automat ma swój stan akceptujący i od tego momentu zaczyna oczekiwać na stan $f_{i+1 \mod k}$.

Jeśli opisany powyżej automat przechodził przez stan akceptujący nieskończenie wiele razy, oznacza to że w pewnym momencie przeszedł do drugiej fazy i każdego ze stanów f_i się doczekał nieskończenie wiele razy. Odpowiada to akceptującemu biegowi $\mathcal{A}$. Z drugiej strony jeśli $\mathcal{A}$ ma akceptujący bieg, opisany automat $\mathcal{B}$ mógł zgadnąć dobry zbiór F_i i dobry moment przejścia do drugiej fazy. Następnie na każdy ze stanów f_i się w końcu doczekał, więc w sumie przechodził przez swój stan akceptujący nieskończenie wiele razy.

4.6 Alternatywna konstrukcja

Niedawno, p. Damian Niwiński zaproponował alternatywny pomysł determinizacji automatów dla słów nieskończonych. Jest to uproszczona i zmodyfikowana konstrukcja Safry. Kluczowy pomysł jest taki, by zamiast przypisywać wierzchołkom w abstrakcyjnym drzewie, zbiory stanów automatu, można odwrócić to przypisanie. Dzięki temu, rozpatrywane pojęcie to funkcja ze zbioru Q w ciągi liczb, odpowiadające położeniu odpowiedniego stanu w drzewie. Obiekt taki nazywać będziemy mapą Safry.

Ustalmy niedeterministyczny automat Büchiego $\mathcal{A} = \langle Q, q_0, \delta, F \rangle$. Oznaczmy przez n liczbę stanów $\mathcal{A}$.

Najważniejszą częścią mapy Safry jest funkcja częściowa $W \colon Q \to \{1, 2, \ldots, 2n\}^*$, o następujących własnościach:

1. obraz $W(Q)$ jest zamknięty ze względu na prefiksy,

2. jeśli jakaś liczba $i \in \{1, 2, \ldots, 2n\}$ występuje w słowach $W(q_1)$ i $W(q_2)$, to słowa te mają wspólny prefiks, zawierający liczbę i,

3. każde ze słów w $W(Q)$ jest różnowartościowe, czyli poszczególne litery występują tam co najwyżej raz.

Dodatkowo, każda mapa Safry niektóre słowa w $W(Q)$, oznacza jako świeże.

Podobnie jak poprzednio, łatwo sprawdzić że wszystkich możliwych map Safry jest skończenie wiele, więc można przyjąć je wszystkie za stany nowego automatu. Jako stan początkowy oznaczmy mapę W_0 która jest wszędzie nie określona, poza stanem q_0 i tam przyjmuje wartość ε.

Pozostaje zdefiniować przejścia konstruowanego automatu. Weźmy dowolną mapę Safry W, oraz literkę a. Przekształceń dokonywać będziemy w kilku krokach:

1. Zdefiniujmy nową mapę Safry W' na stanie $q \in Q$. Weźmy wszystkie stany $p \in Q$, takie że $p \xrightarrow{a} q$. Rozważmy wszystkie ciągi $W(p)$ i wybierzmy z nich alfabetycznie najmniejszy, a spośród porównywalnych najdłuższy, oznaczmy go w. Połóżmy $W'(q) = w$. Dla stanów $q \in Q$, gdzie nie ma żadnego $p \in Q$ spełniającego $p \xrightarrow{a} q$, nowa mapa W' będzie nieokreślona.

2. Dla każdego $q \in F$, zmieńmy wartość $W'(q)$ na wartość $W'(q)i$, gdzie i to pewna litera, nie używana nigdzie indziej w obrazie funkcji W'.

3. Dopóki istnieje takie słowo w nad alfabetem $\{1,2,\ldots,2n\}$, że dla pewnego stanu $q \in Q$ zachodzi $W'(q) > w$ i dla żadnego stanu $p \in Q$ nie zachodzi $W'(p) = w$, zmieniamy wartość $W'(r)$ na w na wszystkich stanach $r \in Q$ spełniających $W'(r) > w$. Można na to patrzeć tak, że jeśli obraz $W'(Q)$ nie jest zamknięty ze względu na prefiksy, to wszystkie słowa, których jakiś prefiks nie leży w $W'(Q)$ skracamy.

4. Wszystkie słowa w których dotyczyły zmiany z poprzedniego punktu (i wciąż są w obrazie) $W'(Q)$ oznaczmy jako świeże, pozostałe oznaczmy jako nie świeże.

Zauważmy, że kolejność wykonywania operacji 3) nie ma znaczenia, a jeśli jakieś słowo w jest oznaczone jako świeże w W', to żadne jego przedłużenie nie leży w obrazie $W'(Q)$.

Powyższa operacja jest deterministyczna, dostajemy więc deterministyczny automat o stanach będących mapami Safry. Oznaczmy go $\mathcal{S}$. Pozostaje zdefiniować warunek akceptacji. Powiemy, że ciąg map Safry W_i jest akceptujący, jeśli istnieje takie słowo $w \in \{1,2,\ldots,2n\}^*$, że od pewnego momentu jest ono zawsze obecne w zbiorze $W_i(Q)$ i nieskończenie często jest świeże. Dość prosto przełożyć to na warunek Mullera (a nawet Rabina).

Pozostaje pokazać, że zdefiniowany powyżej automat jest równoważny $\mathcal{A}$.

Przydatne będą następujące spostrzeżenia:

1. Zbiór stanów na których określona jest odpowiednia mapa Safry, to zbiór stanów osiągalnych w danym momencie ze stanu początkowego.

2. Jeśli dla $i < j$ oraz pewnego stanu $q \in Q$ mamy $W_j(q) = w$, w jest świeże w W_i,W_j i w jest obecne we wszystkich zbiorach $W_i(Q)$,$W_{i+1}(Q),\ldots,W_j(Q)$, to istnieje stan p spełniający $W_i(p) = w$ oraz bieg ze stanu p w momencie i, do stanu q w momencie j, przechodzący przez stan akceptujący.

Ostatnia własność jest kluczowa dla zrozumienia konstrukcji, stanowi ona bezpośredni odpowiednik analogicznego lematu w przypadku zwykłych drzew Safry.

Dowód $L(\mathcal{A}) \subseteq L(\mathcal{S})$. Załóżmy, że istnieje bieg akceptujący $q_0,q_1,\ldots$ automatu $\mathcal{A}$ na danym słowie $\alpha \in A^\omega$. Załóżmy, że mapy Safry uzyskane w trakcie czytania słowa α, to $W_0,W_1,\ldots$. Pokażemy, że istnieje takie słowo $w \in \{1,2,\ldots,2n\}$, które od pewnego momentu jest obecne w obrazach $W_i(Q)$ i nieskończenie wiele razy pewien stan q spełniający $W_i(q) = w$ jest świeży.

Spójrzmy na wartości $W_i(q_i)$. Rozważmy $L = \liminf_i |W_i(q_i)|$. Od pewnego momentu słowa $W_i(q_i)|_L$ się stabilizują na pewnym słowie $H \in \{1,2,\ldots,2n\}^L$, bo gdy prefiks długości L przestanie być kasowany, nie rośnie on leksykograficznie. W takim razie H od pewnego momentu jest stale obecne w $W_i(Q)$. Jednocześnie nieskończenie często q_i jest akceptujący, więc nieskończenie często słowo $W_i(q_i) \geq H$ zostaje wydłużone o dodatkową literkę. Ale po każdym takim wydłużeniu, prędzej czy później długość odpowiednich słów znów wraca do L, więc nieskończenie często H jest świeże. □

Dowód $L(\mathcal{S}) \subseteq L(\mathcal{A})$. Dowód analogiczny jak w przypadku zwykłych drzew Safry:

Załóżmy, że istnieje słowo $w \in \{1,2,\ldots,2n\}^*$, które od pewnego momentu jest stale obecne w $W_i(Q)$ i nieskończenie często jest świeże. Więc istnieją dowolnie długie biegi automatu, które w momentach gdy w jest świeże są mapowane w w przez funkcje W_i. Więc w tych biegach występują dowolnie duże liczby wystąpień stanów akceptujących. Więc korzystając z lematu Königa istnieje bieg, który nieskończenie wiele razy przechodzi przez stan akceptujący. □

4.7 Wnioski

Jeśli w powyższym twierdzeniu zastosujemy konstrukcję bezpośrednią, można w oparciu o dwa powyższe twierdzenia wykazać w prosty sposób zamknięcie NBA na dopełnienia. Możemy mianowicie wziąć dowolny NBA $\mathcal{A}$ rozpoznający język L, rozpatrzyć równoważny mu DMA, tam zanegować formułę boolowską definiującą biegi akceptujące, wreszcie wrócić do NBA, uzyskując automat $\mathcal{B}$ rozpoznający dopełnienie L.

Zadania

Zadanie 4.1. *Przez WMSO, z angielskiego Weak Monadic Second-Order Logic, oznaczamy wariant MSO, w którym wolno kwantyfikować wyłącznie po elementach oraz skończonych podzbiorach uniwersum.*

Pokazać, że każda formuła WMSO($\leq$) na słowach nieskończonych jest równoważna formule logiki MSO($\leq$). *Rozwiązanie na stronie 80.*

Zadanie 4.2. *Rozważmy topologię na zbiorze A^ω, w której bazowymi zbiorami otwartymi są zbiory postaci wA^ω dla $w \in A^*$. Innymi słowy, jest to topologia zadana przez odległość*

$$d(w,v) = \frac{1}{n} \qquad \text{gdzie } n \text{ to pierwsza pozycja, na której różnią się } w,v \in A^\omega.$$

(Niektórzy wolą $1/2^n$ zamiast $1/n$ w powyższym napisie.) Pokazać, że każdy język ω-regularny jest booleowską kombinacją przeliczalnych iloczynów zbiorów otwartych. (Inaczej mówiąc, na poziomie bool(Π_2) w hierarchii borelowskiej.)

Zadanie 4.3. *Wskazać algorytm, który sprawdza czy język ω-regularny (dany przez NBA czy DMA) jest zbiorem otwartym.*

Zadanie 4.4. *Porównać powyższą topologię z topologią zadaną przez odległość:*

$$e(w,v) = \frac{1}{n} \qquad \text{gdzie } n \text{ to najmniejsza ilość stanów w NBA, który akceptuje } w \text{ ale nie } v.$$

Rozwiązanie na stronie 82.

Zadanie 4.5. *Niech $\alpha : A^* \to G$ homomorfizm w grupę. Dla elementów $g,h \in G$ wskazać automat, który akceptuje słowa postaci $\alpha^{-1}(g)(\alpha^{-1}(h))^\omega$.*

Zadanie 4.6. *Wykaż, że języki rozpoznawane przez automaty deterministyczne z warunkiem parzystości, używające priorytetów $\{i, \ldots, j\}$ dla $i \in \{0,1\}$, $j \geq i$ tworzą ścisłą hierarchię.* *Rozwiązanie na stronie 84.*

Rozdział 5

Języki regularne słów skończonych

Spisali: Marek Kiszkis

5.1 Języki regularne słów skończonych a logika pierwszego rzędu (FO)

Podczas tego wykładu ograniczamy się do słów skończonych, i języków składających się ze słów skończonych.

Celem wykładu jest opisanie które języki regularne (słów skończonych) dają się opisać formułami FO, tzn sformułowanie i udowodnienie twierdzenia:

Twierdzenie (v1) Niech L - język regularny. Równoważne są warunki:

1. L opisuje się formułą FO
2. ...

Chcielibyśmy mieć jakiś inny, rozstrzygalny warunek równoważny z (1). Przypomnijmy sobie języki regularne, które nie dają się opisać formułą FO. Przykładami takich języków są: $(aaa)^*, a^{3n}, (abab)^*$ (parzyście wiele razy "ab'').

Zastanówmy się jak wyglądają minimalne automaty deterministyczne (MDF) rozpoznające powyższe języki.

W obu przypadkach możemy zauważyć "cykl po słowie" (w pierwszym automacie istnieje stan q, w którym po 3-krotnym przeczytaniu słowa "a'' wracamy do q, w drugim - istnieje stan r, w którym po 2-krotnym przeczytaniu słowa "ab'' wracamy do r). Bierze się to stąd, że automat pierwszy "liczy" ilość liter "a'' modulo 3, a drugi - ilość wystąpień "ab'' modulo 2.

Intuicyjnie (i bardzo nieformalnie): jeżeli MDF rozpoznający język L cokolwiek "liczy", L nie da się opisać formułą FO.

W celu sformalizowania powyższej hipotezy, wprowadźmy definicję:

Definicja (Automat bezlicznikowy) Automat jest bezlicznikowy, jeśli
$\neg\exists_{q\in Q, w\in A^+, k>1} q \xrightarrow{w^k} q$

Zatem hipoteza na wersję drugą twierdzenia brzmi:

Twierdzenie (v2) Niech L - język regularny. Równoważne są warunki:

1. L opisuje się formułą FO

2. MDF rozpoznający język L jest bezlicznikowy

Istotne jest rozpatrywanie **minimalnego** automatu **deterministycznego** rozpoznającego język L. Jeżeli bowiem rozpatrywalibyśmy automat nieminimalny, mógłby w nim istnieć stan q spełniający powyższy warunek, ale nieosiągalny ze stanu początkowego (a taki q nas nie interesuje). Konieczność rozpatrywania automatu deterministycznego jest oczywista.

Przeprowadzimy dowód (1) $\Rightarrow$ (2) (dokładnie: $\neg(2) \Rightarrow \neg(1)$)

Załóżmy więc, że MDF rozpoznający język L nie jest bezlicznikowy, zatem: $\exists_{q \in Q, w \in A^+, k > 1} q \xrightarrow{w^k} q, q \xrightarrow{w} q' \neq q$

Zauważmy, że stan q musi być osiągalny ze stanu początkowego (gdyby tak nie było, q byłby niepotrzebny, zatem automat nie byłby minimalny), powiedzmy po słowie u. Ponadto stan akceptujący musi być osiągalny z q - gdyby tak nie było, q byłby śmietnikiem, a wiemy że z q możemy dojść do stanu $q' \neq q$.

Istnieje więc słowo v, takie że $uv \in L, uwv \notin L$ lub odwrotnie. Gdyby tak nie było, nie umielibyśmy odróżnić stanów q i q', co przeczy założeniu o minimalności automatu.

Załóżmy teraz, że istnieje formuła φ, opisująca język L. Niech n - maksymalne zagłębienie kwantyfikatorów w φ. Rozpatrzmy słowa $s_1 = u(w^{k^n})v$ i $s_2 = u(w^{k^n+1})v$. Na podstawie powyższych rozważań, $s_1 \in L, s_2 \notin L$ lub odwrotnie. Wynika to z faktu, że po przeczytaniu słowa $u(w^{lk})$ dla dowolnego $l \geq 0$, automat znajduje się w stanie q, zaś po przeczytaniu słowa $u(w^{lk+1})$ - w stanie q'.

Jednak jesteśmy w stanie, za pomocą gry Ehrenfeuchta–Fraïssé pokazać, że formuła φ nie odróżni tych dwóch słów, co kończy dowód.

Implikacji (2) $\Rightarrow$ (1) nie udowodnimy bezpośrednio. Zamiast tego wprowadzimy warunek (3), po czym, udowodnimy (3) $\Rightarrow$ (1) i (2) $\Rightarrow$ (3).

Twierdzenie (v3) Niech L - język regularny. Równoważne są warunki:

1. L opisuje się formułą FO
2. MDF rozpoznający język L jest bezlicznikowy
3. L opisuje się formułą LTL

LTL - logika temporalna, bez zmiennych, jedyne do czego możemy się odwoływać to bieżąca pozycja. Formuły w LTL definiujemy indukcyjnie:

- formuła a jest predykatem sprawdzającym, czy na danej pozycji znajduje się litera a
- jeżeli φ, ψ to formuły LTL to $\varphi \vee \psi$ też jest formułą LTL
- jeżeli φ jest formułą LTL, to $\neg\varphi$ też jest formułą LTL
- jeżeli φ, ψ to formuły LTL, to $\varphi \bigcup \psi$ (φ *until* ψ) też jest formułą LTL, mówiącą: na pewnej pozycji j na prawo od aktualnej pozycji spełniona jest formuła ψ, zaś na wszystkich pozycjach od aktualnej do $(j-1)$-szej spełniona jest formuła φ

Zaś formalnie (po lewej stronie równoważności formuły LTL, po prawej FO lub mniejsze LTL):

- $w \models a \Leftrightarrow w_1 = “a''$
- $w \models \varphi \vee \psi \Leftrightarrow w \models \varphi \vee w \models \psi$
- $w \models \neg\varphi \Leftrightarrow \neg(w \models \varphi)$

- $w \models \varphi \bigcup \psi \Leftrightarrow \exists_{1<i\leq|w|} \ (w_{i..|w|} \models \psi, i \ \forall_{1<j<i} \ w_{j..|w|} \models \varphi)$

  Zauważmy, że powyższa definicja dowodzi implikacji (3) $\Rightarrow$ (1).

  Wprowadzimy jeszcze operator X oraz prawdę, aby pokazać konkretne przykłady formuł LTL.
  $(true \mapsto a \vee \neg a)$
  $\mathrm{X}\varphi \mapsto false \bigcup \varphi$

  Widać, że $w \models \mathrm{X}\varphi \Leftrightarrow w_{2..|w|} \models \varphi$

  Przykładowa formuła w LTL rozpoznająca słowo abc i nic więcej to:
  $a \wedge \mathrm{X} \ (b \wedge \mathrm{X} \ (c \wedge \neg \mathrm{X} \ true))$

Z definicji LTL możemy wyciągnąć jeszcze jeden wniosek: dla danej formuły LTL, odpowiadająca jej formuła w FO nie musi używać więcej niż trzech zmiennych.

Nie jest to prawda w ogólności - np. gdybyśmy rozważali formuły działające na drzewach. Jednak w przypadku formuł działających na słowach, jest to prawda. Zatem możemy sformułować ostateczną już wersję twierdzenia:

Twierdzenie (v4) Niech L - język regularny. Równoważne są warunki:

1. L opisuje się formułą FO
2. MDF rozpoznający język L jest bezlicznikowy
3. L opisuje się formułą LTL
4. L opisuje się formułą FO używającą co najwyżej trzech zmiennych

Implikacja (4) $\Rightarrow$ (1) jest oczywista, implikację (3) $\Rightarrow$ (4) pozostawiamy Czytelnikowi jako proste ćwiczenie.

Pozostała nam do udowodnienia już tylko implikacja (2) $\Rightarrow$ (3) (najtrudniejsza). Udowodnimy ją przez indukcję po dwóch parametrach: po rozmiarze automatu (konkretnie po ilości stanów) i po rozmiarze alfabetu.

Niech A - automat deterministyczny, bezlicznikowy. Nasz cel: dla każdych dwóch stanów p, q w automacie A działającym nad alfabetem Σ, chcemy podać formułę $\varphi_{p,q}^{A,\Sigma}$ (w LTL) akceptującą język $\{w : p \xrightarrow{w} q\}$, tzn język słów po których wczytaniu w stanie p, przejdziemy do stanu q. Najbardziej oczywiście będzie nas interesować formuła $\varphi_{q_i,q_f}^{A,\Sigma}$.

Dla danego słowa w, definiujemy funkcję $w_A : Q \rightarrow Q$ następująco:
jeżeli $q \xrightarrow{w} q'$, to $w_A(q) \leftarrow q'$

Rozważmy 2 przypadki:

1. (baza indukcyjna).
   $\forall_{a\in\Sigma} \ a_A = id$
   wtedy $\varphi_{p,p}^{A,\Sigma} = true$, $\varphi_{p,q}^{A,\Sigma} = false$ dla $p \neq q$
2. (krok indukcyjny).
   $\exists_{a\in\Sigma} \ a_A \neq id$
   Zauważmy, że a_A nie może być permutacją nad Q, bo wtedy automat A nie byłby bezlicznikowy (z pewnego stanu, po słowie jednoliterowym "a'', istniałby cykl długości > 1).
   Zatem $a_A(Q) = P \subset Q$

   Formułę $\varphi_{p,q}^{A,\Sigma}$ napiszemy jako alternatywę trzech formuł:
   $\mathring{\varphi}_{p,q}^{A,\Sigma} \vee \dot{\varphi}_{p,q}^{A,\Sigma} \vee \ddot{\varphi}_{p,q}^{A,\Sigma}$,
   gdzie

(a) $\mathring{\varphi}^{A,\Sigma}_{p,q}$ będzie formułą akceptującą język $\{w : p \xrightarrow{w} q$ i w nie zawiera żadnej litery $a\}$,

(b) $\dot{\varphi}^{A,\Sigma}_{p,q}$ - formułą akceptującą język $\{w : p \xrightarrow{w} q$ i w zawiera dokładnie jedno wystąpienie litery $a\}$,

(c) $\ddot{\varphi}^{A,\Sigma}_{p,q}$ - formułą akceptującą język $\{w : p \xrightarrow{w} q$ i w zawiera ≥ 2 wystąpienia litery $a\}$.

$\mathring{\varphi}^{A,\Sigma}_{p,q} = \varphi^{A,\Sigma\setminus\{a\}}_{p,q}$
Nowy alfabet jest mniejszy, zatem mamy żądaną formułę z założenia indukcyjnego.

$\dot{\varphi}^{A,\Sigma}_{p,q}$ = ("litera a występuje") $\wedge$
($\exists_{r\in Q}$ do litery a jest $\varphi^{A,\Sigma\setminus\{a\}}_{p,r}$,
za literą a jest $\varphi^{A,\Sigma\setminus\{a\}}_{r',q} \wedge r \xrightarrow{a} r'$)

Pierwszy nawias możemy zrealizować formułą: $(a \vee true \bigcup a)$

za literą "a'' jest $\varphi^{A,\Sigma\setminus\{a\}}_{r',q}$ -
idziemy o jeden krok za "a'', i odpalamy $\varphi^{A,\Sigma\setminus\{a\}}_{r',q}$

do litery "a'' jest $\varphi^{A,\Sigma\setminus\{a\}}_{p,r}$ -
będzie oczywiście bazować na $\varphi^{A,\Sigma\setminus\{a\}}_{p,r}$. Jednak w tej formul mogą znaleźć się untile, które w nowym słowie chciałyby "wyjść" za "a''. Musimy więc przerobić tę formułę tak żeby faktycznie dotyczyła ona tylko części słowa znajdującej się przed literą "a'':

Na podstawie φ tworzymy $\hat{\varphi}$, w następujący sposób:

- $\hat{b} = b$
- $\widehat{\varphi \vee \psi} = \hat{\varphi} \vee \hat{\psi}$
- $\widehat{\neg b} = \neg\hat{b} \wedge \neg a$ (żeby dobrze działało na słowie pustym)
- $\widehat{\varphi \bigcup \psi} = (\hat{\varphi} \wedge \neg a) \bigcup (\hat{\psi} \wedge \neg a)$

Formalnie:
Lemat 1. $wav \models \hat{\varphi} \Leftrightarrow w \models \varphi$

Dowód - wprost z definicji daszka.

$\ddot{\varphi}^{A,\Sigma}_{p,q}$ =("są przynajmniej 2a") $\wedge$
($\exists_{r,s'\in Q}$ "do pierwszego a jest $\varphi^{A,\Sigma\setminus\{a\}}_{p,r}$" $\wedge$
"za ostatnim a jest $\varphi^{A,\Sigma\setminus\{a\}}_{s',q}$")
$\wedge$ ("pomiędzy pierwszym a ostatnim a jest dobrze")

Pierwszy nawias łatwo realizujemy podobnie jak poprzednio. Również obie formuły pod kwantyfikatorem konstruujemy z mniejszych, tak jak w poprzednim przypadku.

Pozostaje jeszcze napisać formułę sprawdzającą "środek" słowa.

Podzielmy słowo według wystąpień litery a:

```
....a([]....a)([].......a)([].....a)([]..a)......
   r  r'                                s  s'
```

Każdy fragment

```
([]....a)
```

chcemy potraktować jak pojedynczą literę. Zauważmy, że przy powyższym podziale słowa każdy taki fragment startuje w pewnym stanie ze zbioru P, i kończy w pewnym stanie z P. Możemy więc rozpatrywać te fragmenty jako funkcje $P \to P$, tym samym utożsamiając niektóre słowa, ale to nie szkodzi.

Utwórzmy nowy automat B, jak następuje:

- Stany B - tylko stany ze zbioru P,
- Alfabet $\Gamma = \{f : P \to P\}$
- Przejścia - $p \xrightarrow{f} f(p)$

Oczywiście alfabet automatu B jest najprawdopodobniej o wiele większy niż automatu A, ale B ma mniej stanów (P jest właściwym podzbiotem Q), z założenia indukcyjnego mamy więc formułę $\varphi_{r',s}^{B,\Gamma}$.

Formuła ta jednak działa na słowach nad alfabetem Γ, musimy więc trochę ją przerobić: Każde "f'' w tej formule zamieniamy na:
$\exists_{p \in Q}$ ("jest przejście z p do $f(p)$ kończące się przed najbliższym wystąpieniem litery a")

Wymaga to jeszcze przerobienia tak, żeby untile nie mogły byc prawdziwe w środku "małych" słów, tylko pod koniec fragmentów kończących się na a:
$\widehat{\varphi \bigcup \psi} = (\neg a \vee (a \wedge \mathrm{X}\ \hat{\varphi})) \bigcup (a \wedge \mathrm{X}\ \hat{\psi})$

Konstrukcja formuły $\ddot{\varphi}_{p,q}^{A,\Sigma}$ kończy krok indukcyjny i dowód implikacji (2) $\Rightarrow$ (3).

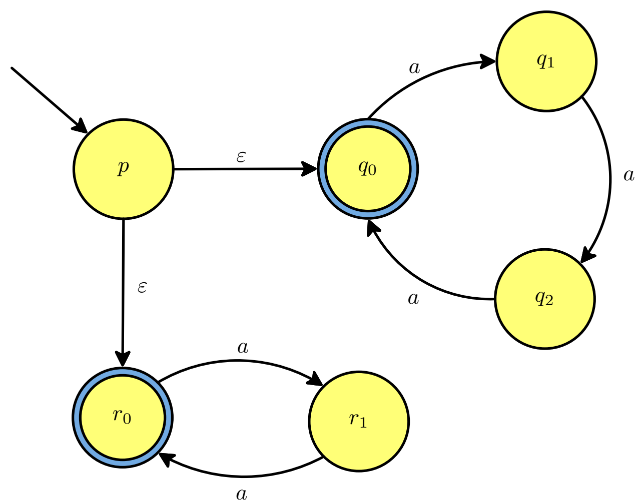
Rozdział 6

Alternacja

Spisali: Michał Pilipczuk i Oskar Skibski

6.1 Alternacja na słowach skończonych

Standardowy automat niedeterministyczny akceptuje słowo, gdy **istnieje** bieg akceptujący. Od teraz będziemy go nazywali także automatem **egzystencjalnym**. Analogicznie możemy rozważać ten sam automat z inną semantyką — automat będzie akceptował słowo, jeśli **każdy** bieg na tym słowie jest akceptujący. Ten automat będziemy nazywać **uniwersalnym**. W ten sposób możemy zdefiniować dwie semantyki tego samego automatu. Zależnie od definicji będzie on akceptował różne języki. Dla automatu egzystencjalnego język ten będziemy nazywać $L_\exists(\mathcal{A})$, dla uniwersalnego $L_\forall(\mathcal{A})$.



Przykład: Automat z rysunku nad alfabetem jednoliterowym, rozważany jako automat egzystencjalny akceptuje słowa o długości podzielnej przez 2 **lub** 3, czyli przystającej do 0,2,3 lub 4 modulo 6. Ten sam automat rozumiany jako automat uniwersalny będzie akceptował jedynie słowa o długości podzielnej przez 2 **i** 3, czyli przez 6.

Nasuwa się pytanie: jakiej długości jest najkrótsze słowo akceptowane przez dany automat, o ile istnieje? Łatwo sprawdzić, że dla automatu egzystencjalnego będzie ono nie większe niż wielkość automatu — gdyż długość najkrótszej ścieżki do stanu akceptującego szacuje się z góry przez liczbę stanów. Da się pokazać, że w automatach uniwersalnych może być ono wykładniczej długości. Wystarczy rozszerzyć przytoczony przykład na więcej liczb pierwszych niż tylko 2 i 3.

Fakt 16. *Problem*

<u>*Dane:*</u>	*Automat uniwersalny* $\mathcal{A}$
<u>*Pytanie:*</u>	*Czy* $L(\mathcal{A}) = \emptyset$*?*

jest PSPACE-zupełny.

W ogólności zachodzi tabela:

	$L_\forall(\mathcal{A})$	$L_\exists(\mathcal{A})$
czy $= \emptyset$?	PSPACE-zupełny	P
czy $= A^*$?	P	PSPACE-zupełny

Problem pustości dla automatu egzystencjalnego oczywiście jest w P, gdyż po prostu za pomocą BFS-a stwierdzamy, czy któryś stan akceptujący jest osiągalny. Z przytoczonego faktu problem pustości dla automatu uniwersalnego jest PSPACE-zupełny. Problem pełności, czyli czy język definiowany przez automat jest całym A^*, jest dualny do problemu pustości poprzez:

Fakt 17. *Jeśli automatowi* $\mathcal{A}$ *przyporządkujemy automat* $\overline{\mathcal{A}}$ *poprzez zamianę każdego stanu akceptującego na nieakceptujący i vice versa, to zachodzi równość*

$$L_\exists(\mathcal{A}) = A^* - L_\forall(\overline{\mathcal{A}})$$

Dowód
Automat egzystencjalny $\mathcal{A}$ akceptuje słowo wtedy i tylko wtedy, gdy istnieje bieg akceptujący, czyli wtedy i tylko wtedy gdy nie każdy bieg jest nieakceptujący, czy wtedy i tylko wtedy gdy automat $\overline{\mathcal{A}}$ nie akceptuje słowa. Stąd słowo należy do języka $L_\exists(\mathcal{A})$, wtedy i tylko wtedy gdy nie należy do języka $L_\forall(\overline{\mathcal{A}})$, co dowodzi tezy. □

Naturalnym zdaje się w tym momencie próba połączenia uniwersalności i egzystencjalności w nową klasę automatów: **automatów alternujących**. Dla każdego stanu powiemy, czy jest on uniwersalny, czy egzystencjalny. W ten sposób zarówno automaty uniwersalne jak i egzystencjalne będą podklasą alternujących, gdzie wszystkie stany są jednego rodzaju.

Definicja 8. ***Automatem alternującym*** *nazywamy szóstkę uporządkowaną*

$$(Q_\exists, Q_\forall, q_0 \in Q, F \subseteq Q, A, \delta \subseteq Q \times A \times Q)$$

gdzie $Q_\exists \cup Q_\forall$ *jest rozłącznym podziałem zbioru* Q *na stany egzystencjalne i uniwersalne,* q_0 *jest stanem początkowym,* F *jest zbiorem stanów akceptujących,* A *jest alfabetem a* δ *jest funkcją przejścia pomiędzy stanami.*

Teraz trzeba będzie zdefiniować co to znaczy, że automat akceptuje słowo $w = a_1 a_2 \ldots a_n$. Pierwsze podejście będzie uogólniało standardową definicję biegu.

Definicja 9. ***Biegiem akceptującym*** *automatu alternującego* $\mathcal{A} = (Q_\exists, Q_\forall, q_0, F, A, \delta)$ *nazywamy ciąg zbiorów stanów* $Q_0, Q_1, \ldots, Q_n$*, takich, że:*

(a) $q_0 \in Q_0$,

(b) $Q_n \subseteq F$,

(c) dla przejścia z Q_{i-1} *do* Q_i *po literze* a_i*:*

- *jeśli* $q \in Q_{i-1} \cap Q_\exists$ *to* ***istnieje*** *tranzycja* $(q, a_i, p) \in \delta$ *dla pewnego* $p \in Q_i$,
- *jeśli* $q \in Q_{i-1} \cap Q_\forall$ *to* ***dla każdej*** *tranzycji* $(q, a_i, p) \in \delta$ *zachodzi* $p \in Q_i$.

Intuicja stojąca za tą definicją jest taka, że Q_i opisuje zbiór stanów z których musimy zaakceptować po wczytaniu i-tej litery. Początkowo musimy zaakceptować co najmniej ze stanu początkowego, na końcu musimy wiedzieć, że stany, w których się potencjalnie znaleźliśmy, są akceptujące. Przejście egzystencjalne odpowiada temu, że automat niedeterministycznie zgaduje stan p, do którego przejdzie. Przejście uniwersalne odpowiada temu, że dla wszystkich potencjalnych przejść musimy umieć dalej zaakceptować.

Powyższa konstrukcja pokazuje też, że automaty alternujące nie mają większej siły wyrazu niż zwyczajne automat niedeterministyczne (czy deterministyczne).

Fakt 18. *Dla każdego automatu alternującego $\mathcal{A}$ istnieje automat niedeterministyczny $\mathcal{B}$ taki, że*

$$L_{alt}(\mathcal{A}) = L_{\exists}(B).$$

Dowód
Automat $\mathcal{B}$ za zbiór stanów będzie miał $P(Q)$, gdzie Q są stanami automatu $\mathcal{A}$. Jego stanem początkowym będzie $\{q_0\}$, stany akceptujące to podzbiory F, zaś przejścia są opisane w definicji biegu akceptującego. □

Z tego faktu płynie następujący, ważny wniosek:

Twierdzenie 19. *Problem*

<u>*Dane:*</u>	*Automat alternujący $\mathcal{A}$*
<u>*Pytanie:*</u>	*Czy $L_{alt}(\mathcal{A}) = \emptyset$?*

jest w PSPACE.

Dowód
Poprzez determinizację maszyn Turinga o wielomianowej pamięci da się udowodnić, że PSPACE=NPSPACE. Wykażemy zatem, że rozważany problem jest w NPSPACE. Istotnie, trzymając w wielomianowej pamięci zbiór stanów automatu $\mathcal{A}$ możemy symulować działanie automatu niedeterministycznego $\mathcal{B}$ z poprzedniego faktu. Dzięki temu, że działamy w NPSPACE możemy symulować jego niedeterminizm. Algorytm zgaduje bieg w automacie $\mathcal{B}$ i w wielomianowej pamięci weryfikuje jego poprawność. □

Warto uczynić uwagę, że problem ten jest oczywiście trudniejszy od PSPACE-trudnego zagadnienia pustości dla automatu uniwersalnego. Stąd jest on też PSPACE trudny, więc w sumie PSPACE-zupełny.

Definicja biegu jest dość niewygodna w użyciu, zwłaszcza we wszelkich dowodach kombinatorycznych. Dlatego wprowadzimy drugą definicję akceptacji słowa, która korzysta z gier.

Grą $G_w^{\mathcal{A}}$ na słowie $w = a_1a_2\ldots a_n$ i automacie alternującym $\mathcal{A}$ nazwiemy grę w której gracze $\exists$ i $\forall$ generują bieg automatu $\mathcal{A}$ na słowie w. Pozycjami gry są pary $(q,i) \in Q \times \{0,1,\ldots,|w|\}$. Gra zaczyna się od pozycji $(q_0,0)$. Ruch wykonuje gracz „kontrolujący” stan — dla stanu egzystencjalnego gracz $\exists$, dla stanu uniwersalnego gracz $\forall$. Ruch polega na zmianie pozycji (q,i) na $(p,i+1)$ pod warunkiem, że $(q,a_{i+1},p) \in \delta$. Czyli gracz, który wykonuje ruch w pozycji (q,i), wybiera jedną z tranzycji automatu. Grę wygrywa gracz $\exists$ jeśli gra dotrze do pozycji $(q,|w|)$ dla $q \in F$, w przeciwnym razie grę wygrywa gracz $\forall$. (Jeśli w pozycji (q,i) nie ma żadnej tranzycji odpowiadającej stanowi q i literze a_{i+1}, to przegrywa gracz kontrolujący q.)

Definicja 10. *Powiemy, że automat $\mathcal{A}$ akceptuje słowo w jeśli gracz $\exists$ w grze $G_w^{\mathcal{A}}$ ma strategię wygrywającą.*

Zanim formalnie udowodnimy równoważność tych dwóch definicji, zobaczmy, że druga z nich jest kombinatorycznie dużo prostsza.

Fakt 20. *Automatowi alternującemu $\mathcal{A}$ przypiszmy automat dualny $\overline{\mathcal{A}}$, w którym wszystkie stany akceptujące zostały zamienione na nieakceptujące i vice versa oraz wszystkie stany uniwersalne zostały zamienione na egzystencjalne i vice versa. Wówczas zachodzi równość*

$$L(\mathcal{A}) = A^* - L(\overline{\mathcal{A}}).$$

Dowód
Zauważmy, że gra $G_w^{\overline{\mathcal{A}}}$ jest dokładnie grą $G_w^{\mathcal{A}}$, tylko zamieniamy graczy rolami. Istotnie, gracze kontrolują

dualne zbiory stanów i na dualnych zbiorach wygrywają, stąd sytuacja wygląda tak, jakby się zamienili. Ze skończoności gry w $G_w^{\overline{\mathcal{A}}}$ gracz $\exists$ ma strategię wygrywającą wtedy i tylko wtedy gdy $\forall$ nie ma, czyli wtedy i tylko wtedy gdy $\exists$ nie ma strategii wygrywającej w $G_w^{\mathcal{A}}$. Stąd automat $\overline{\mathcal{A}}$ akceptuje wtedy i tylko wtedy, gdy $\mathcal{A}$ nie akceptuje co wprost tłumaczy się na tezę. □

W drugiej definicji potrafimy także opisywać zbiory stanów podobne jak w definicji biegu akceptującego.

Fakt 21. *Dla danego automatu alternującego $\mathcal{A}$ istnieje deterministyczny automat lustrzany (czytający słowo od końca) $\mathcal{B}$ o zbiorze stanów $P(Q)$, taki, że $L(\mathcal{A}) = L(\mathcal{B})$.*

Dowód
Automat $\mathcal{B}$ będzie dynamicznie obliczał od końca zbiory stanów, w których $\exists$ ma strategię wygrywającą. Zaczyna ze zbiorem $Q_n = F$ i czytając litery od końca przechodzi od zbioru stanów wygrywających dla sufiksu do zbioru stanów dla sufiksu o jeden dłuższego. Zauważmy, że dla danego zbioru stanów wygrywających Q_i i poprzedniej litery a, poprzedni zbiór stanów wygrywających Q_{i-1} jest wyznaczony deterministycznie. Stan uniwersalny należy doń, jeśli wszystkie przejścia po literze a_i z niego prowadzą do stanów z Q_i. Stan egzystencjalny zaś, jeśli istnieje chociaż jedno takie przejście. Automat akceptuje w stanach zawierających q_0. □

Płynie stąd wniosek:

Fakt 22. *Problem*

Dane:	*Automat alternujący $\mathcal{A}$, słowo w*
Pytanie:	*Czy $w \in L_{alt}(\mathcal{A})$?*

jest w P.

Dowód
Zauważmy, że działanie automatu deterministycznego $\mathcal{B}$ z poprzedniego faktu możemy wielomianowo symulować poprzez algorytm dynamiczny. Trzymając za każdym razem zbiór stanów Q_i i tworzymy zeń przy pomocy a_i zbiór Q_{i-1} zgodnie z opisanymi regułami. Po n krokach otrzymawszy Q_0 sprawdzamy, czy q_0 doń należy. □

Do pełni szczęścia pozostał formalny dowód równoważności obu definicji akceptacji.

Twierdzenie 23. *Definicja akceptacji automatu alternującego przez bieg jest równoważna definicji przez grę.*

Dowód
Wpierw dowiedziemy implikacji ($\Leftarrow$). Załóżmy, że gracz $\exists$ ma strategię wygrywającą w grze. Wówczas, biorąc ciąg zbiorów (Q_i) z dowodu poprzednich faktów, otrzymujemy bieg akceptujący automatu:

- warunek *(a)* jest spełniony, gdyż $\exists$ ma strategię wygrywającą,
- warunek *(b)* jest spełniony, bo $Q_n = F$,
- warunek *(c)* jest spełniony z konstrukcji zbiorów Q_i.

Przejdźmy zatem do implikacji ($\Rightarrow$). Niech (Q_i) będzie biegiem akceptującym. Wówczas $\exists$ będzie poruszał się tak, że dla każdej możliwej pozycji (q,i) w której $G_w^{\mathcal{A}}$ się znajdzie, zajdzie $q \in Q_i$. Ruch gracza $\forall$ oczywiście nie złamie tego warunku, gdyż wszystkie stany do których są przejścia ze stanu uniwersalnego po danej literze słowa należą do kolejnego ze zbiorów Q_i. Jednocześnie gracz $\exists$ może tak grać, by tego warunku nie złamać, gdyż w stanach egzystencjalnych mamy zagwarantowane istnienie przejścia nie wyprowadzającego poza kolejne Q_i. Grając tą strategią gracz $\exists$ zagwarantuje, że dla końcowej pozycji gry (q,n) zajdzie $q \in Q_n$, czyli wobec tego, że $Q_n \subseteq F$, również $q \in F$. □

6.2 Alternacja na słowach nieskończonych

W świetle poprzednich wykładów naturalnym pomysłem zdaje się rozszerzenie pojęcia automatów alternujących na słowa nieskończone. W tym celu musimy przedefiniować pojęcie akceptacji. Wpierw podejdziemy do problemu wykorzystując warunek Mullera, choć później okaże się to nie najlepszym pomysłem.

Zauważmy, że pojęcie gry $G_w^{\mathcal{A}}$ w sposób naturalny przedłuża się na automat nieskończony! Wystarczy, że warunkiem wygranej będzie spełnienie warunku Mullera przez wygenerowany w czasie gry nieskończony ciąg stanów, zamiast po prostu znalezienia się w stanie akceptującym. Oczywiście powstała gra jest grą o niekończonych rozgrywkach, co będzie wkrótce wymagało pewnych wyjaśnień.

Uogólnienie pojęcia biegu wymaga nieco więcej pracy. Tym razem bieg będzie drzewem — intuicyjnie jest to drzewo kodujące strategię gracza $\exists$.

Definicja 11. ***Biegiem akceptującym*** *alternującego automatu z warunkiem Mullera na słowie nieskończonym* $w = a_1a_2\dots$ *jest drzewo* T *etykietowane stanami z* Q *o następującym własnościach:*

(a) *etykieta korzenia to* q_0*,*

(b) *jeśli węzeł o zagłębieniu* i *ma etykietę* $q \in Q_\exists$ *to ma on dokładnie jednego syna o etykiecie* p *takiej, że* $(q,a_{i+1},p) \in \delta$*.*

(c) *jeśli węzeł o zagłębieniu* i *ma etykietę* $q \in Q_\forall$ *to dla każdego* p *takiego, że* $(q,a_{i+1},p) \in \delta$ *ma on syna o etykiecie* p*,*

(d) *na każdej ścieżce nieskończonej idącej w dół drzewa jest spełniony warunek Mullera.*

Oczywiście słowo jest akceptowane, jeśli istnieje dlań bieg akceptujący w sensie powyższej definicji.

Twierdzenie 24. *Definicja akceptacji automatu alternującego na słowach nieskończonych przez bieg jest równoważna definicji przez grę.*

Dowód

Zauważmy, że strategia dla gracza $\exists$ w grze $G_w^{\mathcal{A}}$ przekłada się na budowę biegu akceptującego. Bieg budujemy indukcyjnie — jedyny niedeterminizm w konstrukcji występuje w stanach egzystencjalnych. Wybierzemy tam p tak, jak wybrałby gracz $\exists$ w swojej stategii. Dzięki temu, że strategia jest wygrywająca, warunek Mullera jest spełniony na każdej ścieżce — w przeciwnym razie gracz $\forall$ mółgby pokierować grę zgodnie z tą ścieżką i wygrać grę, podczas gdy $\exists$ gra strategią wygrywającą.

Teraz mając dany bieg skonstruujemy strategię. Pokażemy, że gracz $\exists$ może tak grać, aby ciąg kolejnych stanów odwiedzanych w czasie gry tworzył ścieżkę idącą w dół drzewa będącego biegiem. W stanach egzystencjalnych gracz $\exists$ wybiera ruch zagwarantowany przez warunek *(b)* i w ten sposób przechodzi w dół drzewa. W stanach uniwersalnych dzięki warunkowi *(c)* gracz $\forall$ musi pójść w dół drzewa. Grając tą strategią gracz $\exists$ niezależnie od ruchów $\forall$ doprowadzi do zbudowania z odwiedzanych stanów ścieżki idącej w dół drzewa, na której na mocy warunku *(d)* jest spełniony warunek Mullera. □

Teraz chcielibyśmy przełożyć także inne rezultaty uzyskane na słowach skończonych, przede wszystkim wynik o komplementarności języków akceptowanych przez automat i automat dualny. Tym razem automatowi alternującemu z warunkiem Mullera $\mathcal{A}$ przyporządkujemy automat dualny, w którym warunek Mullera jest zanegowany, zaś wszystkie stany egzystencjalne zamienione na uniwersalne i vice versa. Chcielibyśmy wykazać, że

$$L(\mathcal{A}) = A^\omega - L(\overline{\mathcal{A}}).$$

Próbując powtórzyć dowód skończony łatwo zobaczyć, że wystarczy wykazać, że w grze $G_w^{\mathcal{A}}$ gracz $\exists$ ma strategię wygrywającą wtedy i tylko wtedy, gdy $\forall$ nie ma. Dla gier skończonych stwierdzenie to jest oczywiste. Dla gier nieskończonych wręcz przeciwnie — jest w ogólności nieprawdziwe! (W ogólności znaczy przy niektórych dziwacznych warunkach wygrywających, bardziej złożonych od warunku Mullera.) Mogą się bowiem zdarzyć takie gry, w których żaden z graczy nie ma strategii wygrywającej — dla każdej strategii któregokolwiek z graczy drugi z nich ma strategię lepszą. Aby to zademonstrować, przytoczymy przykład pochodzący od Eryka Kopczyńskiego. Wpierw jednak kilka definicji:

Definicja 12. **Funkcją parzystości** *na zbiorze* Ω *(lub nieskończonym XORem czy też flip setem) nazywamy taką funkcję* $f : 2^\Omega \to \{0,1\}$, *że jeśli* x *i* y *są elementami* 2^Ω *różniącymi się na jednej pozycji, to* $f(x) \neq f(y)$.

Widać, że warunek z definicji poprzez trywialną indukcję przedłuża się do wzajemnej relacji wartości f na ciągach różniących się na skończonej liczbie pozycji. Jeśli różnią się one na parzystej liczbie elementów, to wartości f nań są równe, w przeciwnym razie są różne. Stąd widzimy, że na skończonym Ω istnieją tylko dwie różne funkcje parzystości odpowiadające zadaniu wartości f na dowolnym elemencie i jednoznacznemu przedłużeniu na pozostałe. Intuicja teoriomnogościowa podpowiada jednak:

Fakt 25. *Na zbiorze* Ω *takim, że* $|\Omega| = \kappa \geq \omega$ *istnieje* 2^{2^κ} *funkcji parzystości.*

Dowód
Określmy relację $\backsim$ na 2^Ω taką, że $x \backsim y$ wtedy i tylko wtedy, gdy x i y różnią się na skończonej liczbie pozycji. Oczywiście $\backsim$ jest relacją równoważności. Jednocześnie każda klasa abstrakcji $\backsim$ ma moc $\kappa^{<\omega} = \kappa$, gdyż każdy element pozostający w relacji z danym odpowiada jednoznacznie wyborowi skończonej liczby pozycji na których się różni. Jeśli klas abstrakcji jest λ, to sumując je uzyskujemy, że $\lambda \cdot \kappa = 2^\kappa$. Ponieważ $\kappa < 2^\kappa$ z twierdzenia Cantora oraz $\lambda \cdot \kappa = \max(\lambda,\kappa)$ z twierdzenia Hessenberga, to $\lambda = 2^\kappa$. Korzystając z pewnika wyboru wybieramy reprezentanta z każdej klasy abstrakcji $\backsim$ i określamy wartość f, która jednoznacznie przedłuży się na jego klasę abstrakcji. Jednocześnie łatwo sprawdzić, że każde takie określenie przedłuży się w sposób spójny z definicją funkcji parzystości, gdyż sprawdzamy w niej jedynie elementy należące do jednej klasy abstrakcji. Stąd funkcji parzystości będzie co najmniej tyle, ile sposobów zadania wartości na reprezentantach, czyli 2^{2^κ}, i na pewno nie więcej, gdyż tyle jest też w ogóle funkcji $2^\Omega \to \{0,1\}$. $\square$

Do konstrukcji przykładu naszej gry wystarczy jedna funkcja parzystości f na $\Omega = \omega$, która na mocy właśnie dowiedzionego faktu istnieje. Zauważmy, że funkcję f możemy równie dobrze traktować jako język $L_f = f^{-1}(1) \subseteq \{0,1\}^\omega$.

Nasza gra nazywa się G_f. Gra rozpoczyna się od słowa pustego. Gracze $\exists$ i $\forall$ na przemian dopisują do niego dowolne niepuste, skończone słowa nad alfabetem $\{0,1\}$. Gracz $\exists$ wygrywa grę, jeśli wytworzone słowo nieskończone należy do L_f, w przeciwnym razie wygrywa $\forall$. Grę zaczyna $\exists$.

Twierdzenie 26. *W grze* G_f *żaden z graczy nie ma strategii wygrywającej.*

Dowód
Ideą przewodnią dowodu będzie możliwość „przeskoczenia” ruchu i zamiany graczy rolami.

Załóżmy, że $\exists$ ma strategię wygrywającą. Pokażemy strategię $\forall$, która jest lepsza niż strategia $\exists$. Załóżmy, że $\exists$ zaczął od pewnego słowa v_1. Gracz $\forall$ myśli, co by się stało, gdyby odpowiedział 1. Znając strategię $\exists$ wie, że odpowie on dopisując słowo v_2. Odpowiada więc $0v_2$ i następnie gra strategią $\exists$ zakładającą, że zostało już napisane $v_1 1 v_2$ i ruch należy do gracza $\forall$. Strategia ta oczywiście istnieje, bo po napisaniu $v_1 1$ gracz $\exists$ napisałby v_2. Jednocześnie, grając tą strategią to on wygra, gdyż słowo powstające w wyniku gry po słowie $v_1 1 v_2$ należy do L_f (bo wtedy $\exists$ wygrywa), zatem to samo słowo tylko z zamienionym $v_1 1 v_2$ na $v_1 0 v_2$ nie należy.

Załóżmy teraz, że $\forall$ ma strategię. Podobnie jak poprzednio pokażemy lepszą strategię dla $\exists$. Patrzy on, jakie słowo v_1 odpowiedziałby $\forall$, gdyby $\exists$ rozpoczął od 1. Rozpoczyna więc grę od słowa $0v_1$ i przyjmuje strategię $\forall$ zakładającą, że zostało skonstruowane już słowo $1v_1$ oraz $\exists$ ma ruch. Argument podobny do poprzedniego pokazuje, że przyjmując tę strategię $\exists$ zawsze wygrywa. $\square$

6.3 Determinacja gier nieskończonych a automaty alternujące

W dalszej części tego rozdziału zastanowimy się kiedy gra jest zdeterminowana, czyli dla jakich warunków akceptowania zawsze któryś z graczy ma strategię wygrywającą. W tym celu najpierw bardziej formalnie zdefiniujemy pojęcie gry oraz strategii.

Definicja 13. ***Grą*** *(nieskończoną) nazywamy trójkę uporządkowaną*

$$(V = V_\exists \cup V_\forall, E \subseteq V \times V, Win_\exists \subseteq V^\omega)$$

gdzie V jest wierzchołkami graczy ($V_\exists$ gracza $\exists$, $V_\forall$ gracza $\forall$), E jest zbiorem krawędzi między wierzchołkami, a $Win_\exists$ jest warunkiem zwycięstwa gracza $\exists$. Wierzchołki z krawędziami nazywamy czasem ***areną****.*

Rozgrywką *nazywamy ciąg wierzchołków $v = v_1v_2v_3... \in V^\omega$ taki że $\forall_{i>0}(v_i, v_{i+1}) \in E$. Gracz $\exists$* ***wygrywa*** *rozgrywkę v kiedy $v \in Win_\exists$.*

Dla dalszych rozważań zakładamy że z każdego wierzchołka wychodzi co najmniej jedna krawędź. Warto zwrócić też uwagę na to, że nie wskazujemy wierzchołka początkowego - gra może zacząć się w dowolnym miejscu.

Definicja 14. ***Strategią*** *dla gracza $\exists$ nazywamy funkcję $\sigma_\exists \colon V^\star V_\exists \to V$, przy czym jeżeli $\sigma_\exists(v_1v_2...v_n) = v_{n+1}$ to $(v_n, v_{n+1}) \in E$.*

Strategię nazywamy ***bezpamięciową*** *kiedy nie zależy od wcześniejszego przebiegu rozgrywki, a jedynie od stanu w jakim się znajduje. Można ją wówczas zdefiniować jako $\sigma_\exists \colon V_\exists \to V$.*

Powiemy że ***strategia*** *$\sigma_\exists$* ***wygrywa*** *z wierzchołka v_i jeśli dla każdej strategii $\sigma_\forall$ rozgrywka wyznaczona przez strategie jest zwycięska dla gracza $\exists$.* ***Gracz*** *$\exists$* ***wygrywa*** *z wierzchołka v_i jeśli ma w tym wierzchołku strategię wygrywającą.*

Wszystkie powyższe definicję analogicznie defniujemy dla gracza $\forall$.

Gra jest ***zdeterminowana*** *w v_i jeśli $\exists$ albo $\forall$ ma strategię wygrywającą w v_i.*

Aby wprowadzić twierdzenie Martina będziemy musieli przypomnieć czym są zbiory borelowskie.

Definicja 15. ***Zbiorem otwartym*** *nazywamy zbiór, w którym dla każdego jego punktu istnieje otoczenie w całości zawarte w tym zbiorze. Zbiór słów nieskończonych jest otwarty kiedy to czy słowo należy do zbioru zależy wyłącznie od jego prefisku (zbiory te są zatem postaci $\bigcup_{i\in\mathbb{I}} w_i\{0,1\}^\omega$ gdzie $w_i \in V^\star$.*

Dopełnienie zbioru otwartego nazywamy ***zbiorem domkniętym****.*

Zbiorem borelowskim *nazywamy każdy element najmniejszego ciała zawierającego wszystkie zbiory otwarte. Wynika z tego, że zbiór będący przeliczalną sumą / iloczynem przeliczalnych sum / iloczynów, ..., zbiorów otwartych jest zbiorem Borelowskim.*

Twierdzenie 27. *(Martina) Rozważmy grę, gdzie $Win_\exists$ jest zbiorem borelowskim. Wówczas gra jest zdeterminowana w każdym wierzchołku.*

To twierdzenie pozostawimy bez dowodu, w późniejszej części tej pracy udowodnimy za to twierdzenie mocniejsze którego to będzie bezpośrednią konsekwencją.

Spróbujemy teraz skorzystać z twierdzenia Martina dla gry automatowej $G_w^{\mathcal{A}}$ zdefiniowanej w poprzednim rozdziale. Udowodnijmy najpierw mniejszy fakt - że warunek wygrywający jest borelowski.

Fakt 28. *Warunek wygrywający w grze automatowej $G_w^{\mathcal{A}}$ jest borelowski.*

Dowód. Zdefiniujmy grę automatową zgodnie z nową formalną definicją gry. Zbiorem wierzchołków są pary (stan, numer ruchu) (czyli $V = Q \times \mathbb{N}$), przy czym $V_\exists = Q_\exists \times \mathbb{N}$ i $V_\forall = Q_\forall \times \mathbb{N}$. Zbiór krawędzi E składa się z par $(q, i) \times (p, i + 1)$ o ile w $\mathcal{A}$ jest przejście z q do p po literze a_{i+1} (zaczynamy ze stanu $(q, 0)$ wczytując literę a_1). Warunek zwycięstwa gracza $\exists$ - $Win_\exists \subseteq (Q \times \mathbb{N})^\omega$ - składa się zatem z wszystkich ciągów krotek, dla których $q_0q_1q_2...$ spełnia warunek Mullera z $\mathcal{A}$.

Czemu ten warunek jest borelowski? $Win_\exists$ jest boolowską kombinacją warunków "stan q występuje nieskończenie często", z kolei ten warunek możemy zapisać jako $\bigcap_{i\in\mathbb{N}} \bigcup_{j\geq i}$ "stan q występuje na pozycji j". Skoro środek tego wyrażenia jest zbiorem otwartym, to całe wyrażenie (jako iloczyn sum zbiorów otwartych) jest zbiorem bolerowskim. □

Korzystając z powyższego faktu oraz twierdzenia Martina udowodnimy teraz nowy fakt.

Fakt 29. *Niech $\mathcal{A}$ będzie automatem alternującym z warunkiem Mullera. Wówczas dla $w \in A^\omega$ zdania "w jest akceptowane przez $\mathcal{A}$" i "w nie jest akceptowane przez $\overline{\mathcal{A}}$" są równoważne.*

Dowód. Fakt ten pokażemy poprzez ciąg zdań równoważnych:

(a) w jest akceptowane przez $\mathcal{A}$

(b) $\exists$ wygrywa grę $G_w^{\mathcal{A}}$

(c) $\forall$ wygrywa grę $G_w^{\overline{\mathcal{A}}}$

(d) $\exists$ nie wygrywa gry $G_w^{\overline{\mathcal{A}}}$

(e) w nie jest akceptowane przez $G_w^{\overline{\mathcal{A}}}$

Równoważność zdań (a) i (b) jest oczywista i wynika z definicji gry automatowej $G_w^{\mathcal{A}}$.

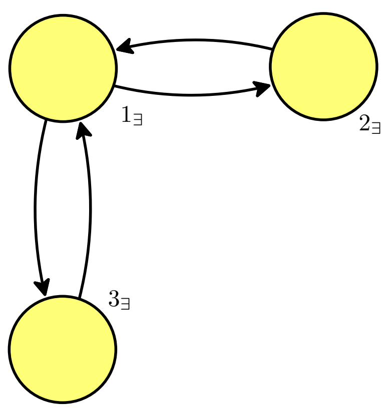
Równoważność zdań (b) i (c) wynika natomiast z definicji automatu dualnego. Skoro $\exists$ wygrywa w $G_w^{\mathcal{A}}$, ma jakąś strategię wygrywającą $\sigma_\exists$. W grze dualnej $\forall$ grając analogiczną strategią $\overline{\sigma_\forall}$ wygrywa - gdyby nie wygrywał strategię pokonującą go można by było przenieść na grę oryginalną.

Wynikanie zdanie (d) z (c) jest oczywista, natomiast w drugą stronę musimy skorzystać z twierdzenia Martina - możemy, bo warunek wygrywający jest borelowski, co udowodniliśmy powyżej.

W końcu równoważność zdań (d) i (e) wynika z definicji gry. □

Następne twierdzenie jakie zdefiniujemy mówi, że dla pewnych warunków akceptacji dla każdego wierzchołka któryś z graczy ma bezpamięciową strategię wygrywającą (Tw. Martina mówiło też o strategiach z pamięcią). Zastanówmy się najpierw czy warunek Mullera gwarantował prawdziwość tego zdania.

Weźmy następującą arenę z warunkiem Mullera mówiącym że stany $2_\exists$ oraz $3_\exists$ występują nieskończenie często.



Łatwo zauważyć, że gracz $\exists$ grając strategią bezpamięciową za każdym razem kiedy znajdzie się w wierzchołku $1_\exists$ pójdzie w tą samą stronę - do wierzchołka $2_\exists$ lub $3_\exists$ - czym spowoduje że drugi wierzchołek wystąpi skończoną ilość razy! Nie istnieje zatem bezpamięciowa strategia wygrywająca dla $\exists$, przy czym istnieje oczywiście strategia wykorzystująca pamięć - wystarczy zapamiętać gdzie poszliśmy z wierzchołka $1_\exists$ i kiedy się w nim znowu znajdziemy skręcić w drugą stronę.

Twierdzenie 30. *Niech G będzie będzie grą w której warunek akceptacji jest warunkiem parzystości. Wówczas dla każdego wierzchołka v jeden z graczy ma bezpamięciową strategię wygrywającą.*

Zanim przejdziemy do dowodu przypomnimy czym jest warunek parzystości oraz zdefiniujemy pomocniczy fakt o łączeniu strategii.

Definicja 16. *Określmy dodatkową funkcję* $\Omega\colon V \to \{0, 1, ...n\}$ *przypisującą każdemu wierzchołkowi pewną liczbę naturalną.* ***Warunek parzystości*** *spełniają wówczas tylko te ciągi* $v_1 v_2 ...$*, dla których* $\limsup \Omega(v_i)$ *jest parzyste.*

Warunek parzystości możemy zdefiniować także określając porządek liniowy na wierzchołkach i zbiór $F \subseteq V$. Wówczas warunek parzystości spełniają te ciągi, dla których największy wierzchołek występujący nieskończenie często należy do zbioru F.

Fakt 31. *Załóżmy że dla* $W \subseteq V$ *w każdym wierzchołku* $w \in W$ *gracz* $\exists$ *ma strategię wygrywającą* $\sigma_{\exists,w}$*. Niech* $win_\exists(\sigma_\exists)$ *będzie zbiorem wierzchołków z których wygrywa* $\exists$ *grając zgodnie z* $\sigma_\exists$*. Wówczas istnieje strategia* $\sigma_\exists$ *taka, że* $W \subseteq win_\exists(\sigma_\exists)$*.*

Dowód. Postaramy się ze strategii dla wierzchołków zbioru W skonstruować wspólną strategię zdefiniowaną na całym zbiorze W. W tym celu ponumerujmy strategie $\sigma_{\exists,w}$ kolejnymi liczbami naturalnymi (równie dobrze możemy zdefiniować dobry porządek na W). Czym będzie nasze $\sigma_\exists(v)$?

Niech $X_v = \{w \in W : v \in win(\sigma_{\exists,w})\}$, czyli niech będzie zbiorem wszystkich wierzchołków z W takich, że ich strategia $(\sigma_{\exists,w})$ jest zdefiniowana dla wierzchołka v. Niech $x_v = inf(X)$, czyli będzie najmniejszym takim wierzchołkiem (zgodnie ze zdefiniowanym porządkiem). Wówczas naszą uogólnioną strategię $\sigma_\exists$ w wierzchołku v określamy zgodnie ze strategią z tego wierzchołka - $\sigma_\exists(v) = \sigma_{\exists,x_v}(v)$.

Dlaczego nasza konstrukcja działa? Łatwo zauważyć, że grając strategią $\sigma_\exists$ z każdym kolejnym ruchem numer strategii jaką naśladujemy może się tylko zmniejszyć - jeżeli wykonaliśmy ruch zgodny ze strategią o numerze i to w wierzchołku w którym się znaleźliśmy ta strategia jest także zdefiniowana. Skoro tak, to po skończonej ilości ruchów będziemy naśladować tę samą wygrywającą strategię. □

Możemy teraz przejść do dowodu twierdzenia.

Dowód. Dowód przeprowadzimy przez indukcję po n, gdzie n jest maksymalną rangą funkcji parzystości. Załóżmy także, że n jest parzyste (czyli "dobre" dla $\exists$) - gdyby nie było w dalszym dowodzie wystarczyłoby zamienić graczy $\exists$ i $\forall$ miejscami.

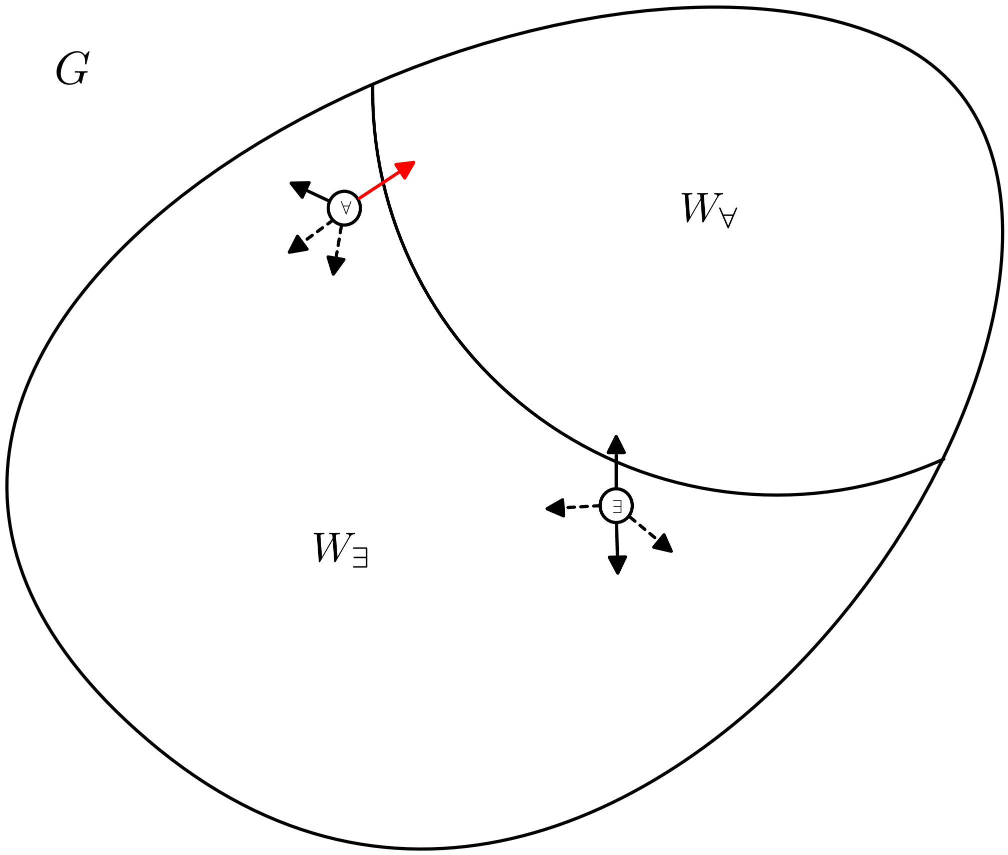
Podzielimy teraz wierzchołki na dwa zbiory: zbiór $W_\forall$ z wierzchołkami z których $\forall$ ma pozycyjną strategię wygrywającą oraz $W_\exists$ - z resztą. Naszym celem jest zatem pokazanie, że dla każdego wierzchołka z $W_\exists$ gracz $\exists$ ma pozycyjną strategię wygrywająca.

Najpierw zauważmy, że skoro $\forall$ ma strategię pozycyjną z każdego wierzchołka $W_\forall$ to z faktu udowodnionego powyżej wiemy, że ma pewną uogólnioną strategię pozycyjną $\sigma_\forall$ która wygrywa ze wszystkich wierzchołków tego zbioru. Wiemy też, że nie opuścimy zbioru $W_\forall$ jeżeli w nim zaczniemy:

- gdyby gracz $\exists$ wykonał ruch poza obszar $W_\forall$ po którym mógłby nie przegrać, wówczas wierzchołek z którego wykonał ten ruch nie może należeć do $W_\forall$ - gracz $\forall$ nie ma z niego strategii wygrywającej! Gdyby natomiast mógł wyjść, ale i tak by przegrał, wówczas wierzchołek do którego wychodzi także powinien należeć do $W_\forall$ - sprzeczność

- gdyby gracz $\forall$ wykonujący ruchy zgodnie ze strategią $\sigma_\forall$ wyszedł poza $W_\forall$ to oznaczałoby, że z wierzchołka do jakiego wyszedł także ma strategię pozycyjną - sprzeczność

Rozpatrzmy teraz jakie krawędzie mogą wychodzić z wierzchołków w zbiorze $W_\exists$.



Czerwone strzałki oznaczają krawędzie które nie mogą wystąpić.

Na diagramie zaznaczone zostały trzy proste, ale ważne dla dalszego dowodu, uwagi. Po pierwsze nie może istnieć wierzchołek gracza $\forall$ z którego jakakolwiek krawędź prowadziłaby do zbioru $W_\forall$ - wówczas wierzchołek ten także powinien należeć do tego zbioru, jako że gracz ma pozycyjną strategię wygrywająca z niego - wejście do $W_\forall$ i dalej granie zgodnie z $\sigma_\forall$. Po drugie, skoro z wierzchołków $\forall$ nie ma krawędzi do $W_\forall$, to zgodnie z założeniem poczynionym odnośnie gier musi istnieć jakaś krawędź prowadząca do $W_\exists$. Po trzecie, także każdy wierzchołek gracza $\exists$ musi mieć co najmniej jedną krawędź prowadzącą do $W_\exists$ - gdyby jej nie było wówczas z tego wierzchołka gra przeniosłaby się do $W_\forall$ i by przegrał, czyli byłby to kolejny wierzchołek zgodny z definicją $W_\forall$.

Z powyższych uwag wynika, że każdy wierzchołek w $W_\exists$ ma krawędź prowadzącą do innego wierzchołka w $W_\exists$. Możemy zatem określić grę obciętą do wierzchołków z tego zbioru. Zanim zajmiemy się nią, zdefiniujemy pojęcie atraktora które przyda nam się w dalszej części dowodu.

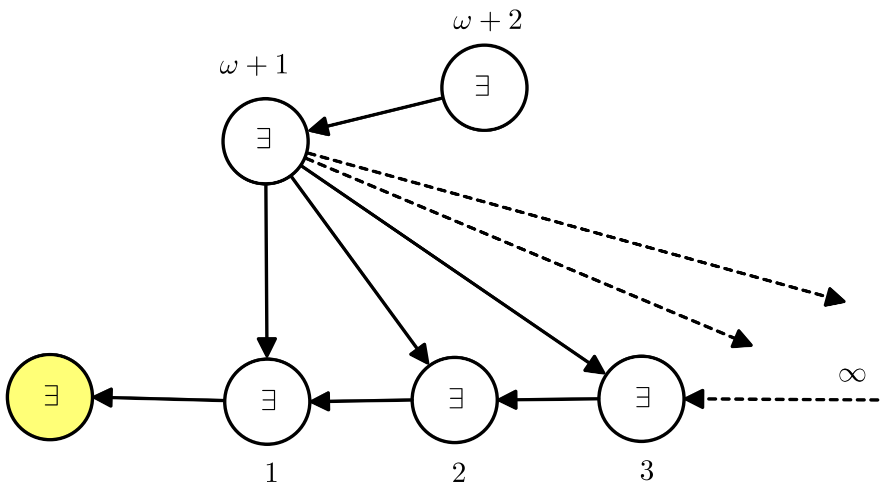
Definicja 17. *Niech H będzie grą, a U pewnym zbiorem wierzchołków. Wówczas* ***atraktor*** *dla gracza $\exists$ (zapisywany jako $Attr_\exists(U)$) jest pierwszym punktem stabilizacji funkcji $Attr_\exists^\alpha(U)$ zdefiniowanej następująco:*

- $Attr_\exists^0(U) = U$

- $Attr_\exists^{\alpha+1}(U) = Attr_\exists^\alpha(U) \cup \{v \in W_\exists :$ *istnieje krawędź z v prowadząca do* $Attr_\exists^\alpha(U)\} \cup \{v \in W_\forall$*: każda krawędź z v prowadzi do* $Attr_\forall^\alpha(U)\}$

- $Attr_\exists^\alpha(U) = \bigcup_{\beta<\alpha} Attr_\exists^\beta(U)$ *dla przypadków granicznych*

Funkcja $Attr_\exists^\alpha$ zdefiniowana jest nie tylko dla liczb naturalnych ale dla wszystkich liczb porządkowych. Dlaczego jest to potrzebne możemy zobaczyć na poniższym przykładzie.



Zbiorem U jest wierzchołek zaznaczony na żółto, a wierzchołków w ciągu jest nieskończenie wiele. Przy wierzchołkach zapisane są indeksy pierwszych zbiorów $Attr_{\exists}^{\alpha}$ do których one wpadają - wierzchołki "górne" wpadają do zbioru ciągu atraktorów dopiero po nieskończenie wielu iteracjach powiększania.

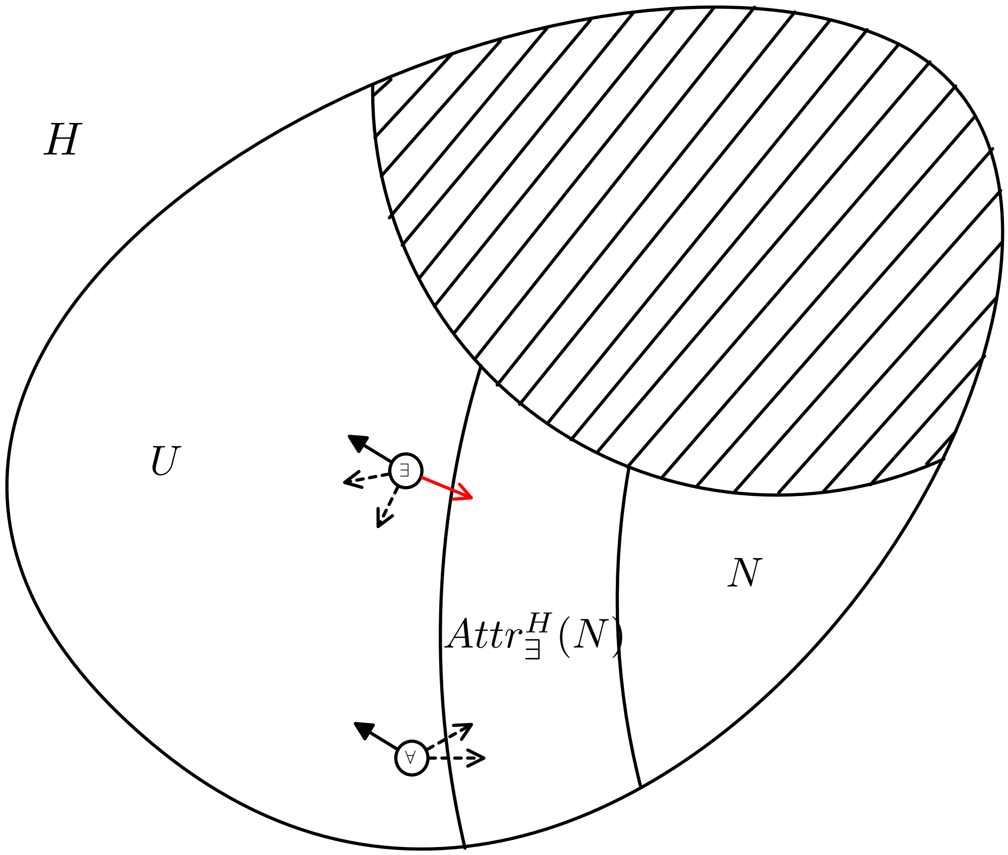
Zdefiniujmy teraz strategię atraktorową.

Definicja 18. *Określmy pozycyjną strategię w oparciu o zbiory $Attr_{\exists}^{\alpha}(U)$ użyte w definicji atraktora dla wierzchołka $v_{\exists} \in Attr_{\exists}(U)$. Jeżeli $v_{\exists} \in U$, to strategię definiujemy jakkolwiek. Jeżeli nie, to musi istnieć takie najmniejsze β, że $v_{\exists} \in Attr_{\exists}^{\beta+1}$ i $v_{\exists} \notin Attr_{\exists}^{\beta}$. Skoro tak, to wierzchołek $v_{\exists}$ wpadł do zbioru, ponieważ jakaś krawędź z niego prowadzi do zbioru $Attr_{\exists}^{\beta}$. Niech zatem nasza strategia prowadzi gracza ∃ tą krawędzią. Tak zdefiniowaną strategię nazywamy* ***strategią atraktorową****.*

Fakt 32. *Jeśli gracz ∃ gra zgodnie ze strategią atraktorową i gra zaczyna się w atraktorze, to w skończonej liczbie kroków wyląduje w zbiorze U.*

Dowód. Rozważmy rozgrywkę v_1,v_2,v_3,v_4,... aż do uzyskania wierzchołka z U zgodną ze strategią atraktorową. Oznaczmy przez α_i indeks pierwszego elementu ciągu $Attr_{\exists}^{\alpha}$ zawierającego wierzchołek v_i. Jeżeli wierzchołek v_i jest wierzchołkiem gracza ∃ to zgodnie ze strategią atraktorową α_{i+1} jest mniejszy o jeden. Jeśli natomiast v_i jest gracza ∀, to zgodnie z definicją atraktora wszystkie krawędzie z tego wierzchołka prowadzą do któregoś z wcześniejszych zbiorów $Attr_{\exists}^{\alpha}$. Udowodniliśmy w ten sposób, że ciąg α_i jest monotonicznie malejący. Pozostaje dowieść, że rozgrywka jest skończona, czyli że zbiór U osiągniemy w skończonej ilości kroków. [niejasne] □

Wróćmy zatem do naszego dowodu. Rozważmy teraz nową grę obciętą do wierzchołków $W_{\exists}$: $H = G|W_{\exists}$ (powiedzieliśmy już że jest ona dobrze zdefiniowana). Niech N będzie zbiorem wierzchołków o randze n w zbiorze $W_{\exists}$. Jak pokażemy później, gracz ∃ w jakimś sensie będzie chciał wpadać do wierzchołków o randze n (z $W_{\exists}$, $W_{\forall}$ jest dla gracza ∃ nieporządane), wyznaczmy zatem atraktor zbioru N: $Attr_{\exists}^{H}(N)$. Przeanalizujmy tą sytuację.



Sytuacja jest dość podobna do tej z poprzedniego diagramu, tylko tym razem zamiast zbioru $W_\forall$ "złego" dla gracz $\exists$, mamy zbiór atraktor $Attr_\exists^H(N)$ "zły" dla gracza $\forall$. Rozpatrzmy krawędzie wychodzące z wierzchołków zbioru U. Jeżeli jest to wierzchołek gracza $\exists$ to oczywiście nie może istnieć krawędź do atraktora, czyli musi istnieć co najmniej jedna krawędź do zbioru U. Natomiast jeżeli jest to wierzchołek $\forall$, to także musi on posiadać strzałkę do U - gdyby wszystkie prowadziły do atraktora ten wierzchołek także zawierałby się w atraktorze! Możemy zatem na zbiorze U określić nową grę - oznaczmy ją jako F - i nareszcie użyć założenia indukcyjnego, ponieważ w zbiorze U nie ma rangi n. Wiemy więc, że zbiór U możemy podzielić na dwa zbiory - zbiór $U_\exists$ w którym z każdego wierzchołka pozycyjną strategię wygrywającą ma gracz $\exists$ oraz $U_\forall$ w którym z każdego wierzchołka taką strategię ma $\forall$. Ponownie łącząc strategie gracza $\exists$ otrzymujemy jedną pozycyjną strategię wygrywającą $\sigma_\exists^U$ zdefiniowaną na zbiorze $U_\exists$.

Zastanówmy się teraz nad zbiorem $U_\forall$. Skoro z każdego wierzchołka gracz $\forall$ ma strategię wygrywającą w grze F, to czy ma ją także w grze G? Czyli czy gracz $\exists$ w grze G może opuścić zbiór $U_\forall$? Gracz $\exists$ nie ma oczywiście krawędzi do $U_\exists$, bo stamtąd by wygrał. Nie może też mieć krawędzi do atraktora, co stwierdziliśmy powyżej. Pozostaje mu tylko ewentualna krawędź do zbioru $W_\forall$ w którym i tak przegrywa! Oznacza to zatem, że strategia pozycyjna w $U_\forall$ jest dobra też w grze G, czyli tak naprawdę $U_\forall$ powinno zawierać się w $W_\forall$. Musi zatem być puste. Wiemy więc, że gracz $\exists$ ma strategię określoną na całym zbiorze U.

Kończąc dowód zdefiniujmy strategię pozycyjną $\sigma_\exists$ w całej grze G (wygrywającą dla wszystkich wierzchołków $W_\exists$):

- dla $w \in W_\forall$ idziemy gdziekolwiek

- dla $w \in N$ idziemy gdziekolwiek poza $W_\forall$

- dla $w \in Attr_\exists^H(N) \setminus N$ $\sigma_\exists$ zachowuje się zgodnie ze strategią atraktorową

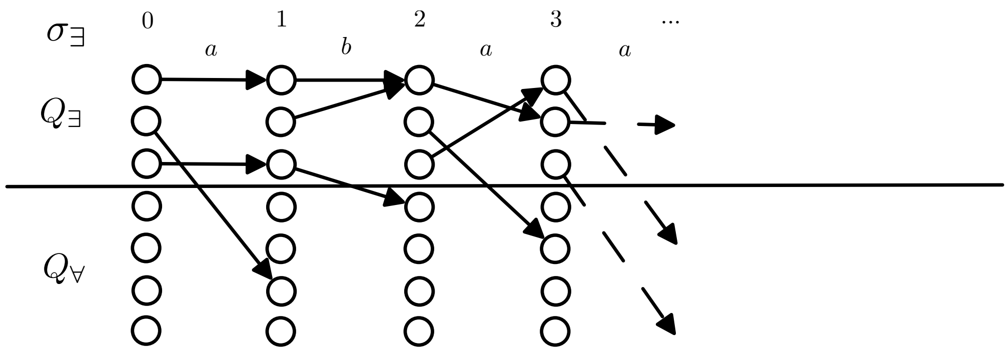
- dla $w \in U$ $\sigma_\exists$ zachowuje się zgodnie z $\sigma_\exists^U$

Dlaczego ta strategia przyniesie graczowi $\exists$ sukces? Rozgrywka zaczęta z poza $W_\forall$ tam nie wejdzie - $\exists$ unika ruchów do $W_\forall$, a $\forall$ nie ma takich ruchów. Jeżeli rozgrywka nieskończenie często zahacza o zbiór N to gracz $\exists$ wygrywa - najwyższa ranga jest parzysta. Jeżeli nie, oznacza to, że gra od pewnego momentu toczy się w U, a na tym zbiorze gracz $\exists$ ma zdefiniowaną strategię wygrywającą. □

Z powyższego twierdzenia możemy wyciągnąć następujący wniosek o ekspresywności automatów alternujących z warunkiem parzystości:

Fakt 33. *Dla każdego automatu alternującego z warunkiem parzystości istnieje równoważny automat niedeterministyczny z warunkiem Büchiego (NBA).*

Dowód. Niech $\mathcal{A}$ będzie naszym automatem alternującym z warunkiem parzystości. Z definicji automatu alternującego wiemy, że $\mathcal{A}$ akceptuje słowo, o ile gracz $\exists$ wygrywa grę automatową $G_w^\mathcal{A}$. Przypomnijmy że wierzchołkami gry automatu $\mathcal{A}$ ze stanami Q są krotki $(Q \times \mathbb{N})$, natomiast gra zaczyna się w stanie $(q_0, 0)$ dla q_0 - stanu początkowego automatu.



Skoro automat ma warunek parzystości, to wiemy że z każdego wierzchołka któryś gracz ma pozycyjną strategię wygrywającą. Oznacza to zatem, że aby słowo w było akceptowane gracz $\exists$ musi mieć pozycyjną strategię wygrywającą ze stanu startowego. Czym jest strategia pozycyjna w naszej grze $G_w^\mathcal{A}$? Dla każdego swojego wierzchołka, czyli (q, i) dla $q \in Q_\exists$, gracz $\exists$ musi wybrać jedną z krawędzi. Krawędzie prowadzą do wierzchołków $(q, i+1)$ dla $q \in Q_\exists \cup Q_\forall$, zatem na każdym poziomie dla każdego stanu z $Q_\exists$ gracz $\exists$ automatu wybieramy jeden dowolny stan automatu (niekoniecznie swój). Strategię możemy więc zakodować jako słowo nieskończone $[\sigma_\exists] \in (Q^{Q_\exists})^\omega$. Analogicznie strategię gracza $\forall$ możemy zakodować jako $[\sigma_\forall] \in (Q^{Q_\forall})^\omega$.

Pokażemy teraz, że zdanie "istnieje pozycyjna strategia $\sigma_\exists$, która wygrywa z każdą pozycyjną strategią $\sigma_\forall$" jest rozpoznawalne przez automat niedeterministyczny z warunkiem Büchiego, co zakończy nasz dowód.

Najpierw rozważmy język $L_1 = \{w \otimes [\sigma_\exists] \otimes [\sigma_\forall]$: w grze $G_w^\mathcal{A}$ pozycyjna strategia $\sigma_\exists$ wygrywa z pozycyjną strategią $\sigma_\forall\}$ $(L_1 \subseteq (A \times Q^{Q_\exists} \times Q^{Q_\forall})^\omega)$. Nie powinno być problemem stworzenie deterministycznego automatu, który symulując grę na $\mathcal{A}$ sprawdza czy finalna ścieżka jest wygrywająca dla $\exists$.

Rozważmy teraz dwa kolejne języki - L_2 oraz docelowy L_3:
$L_2 = \{w \otimes [\sigma_\exists]$: w grze $G_w^\mathcal{A}$ pozycyjna strategia $\sigma_\exists$ wygrywa z każdą pozycyjną strategią $\sigma_\forall\}$.
$L_3 = \{w$: w grze $G_w^\mathcal{A}$ istnieje pozycyjna strategia $\sigma_\exists$, która wygrywa z każdą pozycyjną strategią $\sigma_\forall\}$.

Załóżmy, że L_2 jest regularny i że możemy go zapisać formułą MSO φ_2. Wówczas język L_3 możemy (mało formalnie) zapisać formułą $\exists_{\sigma_\exists}\varphi_2$, czyli L_3 jest rzutem języka L_2 - a języki regularne są zamknięte na rzuty. Upewnijmy się zatem, że L_2 jest językiem regularnym. Rozpatrzmy relację pomiędzy L_2 a regularnym L_1 - jeżeli L_1 zapiszemy jako formułę φ_1, to L_2 możemy zapisać jako $\neg\exists\sigma_\forall\neg\varphi_1$, czyli słowami "nie istnieje taka pozycyjna strategia gracza $\forall$, że formuła φ_1 nie jest spełniona". Czyli tym razem mamy dopełnienie rzutu dopełnienia języka regularnego. Zatem L_2 też jest regularny, co kończy nasz dowód. □

Zadania

Słowa skończone.

Zadanie 6.1. *Pokazać, że problem niepustości jest PSPACE-trudny dla automatów alternujących na słowach skończonych.*

Zadanie 6.2. *Pokazać, że dla każdej formuły LTL istnieje równoważny automat alternujący (z ϵ-przejściami) rozmiaru liniowego.*

Zadanie 6.3. *Pokazać, że wzrost automatu alternującego przy rzutowaniu potrafi być gorszy niż wielomianowy. Wskazówka: niepustości dla MSO(s) nie można rozstrzygać w czasie wykładniczym.*

Zadanie 6.4. *(MB: Nie zastanawiałem się nad tym). Alfabet jest jednoliterowy. Ile potrzeba stanów w automacie alternującym, żeby zdefniować język $\{a^{2^n}\}$?*

Zadanie 6.5. *Rozważmy automaty alternujące dwukierunkowe. To znaczy, przejścia są typu: „jeśli stan to p, litera to a, pozycja jest (pierwsza/ostatnia/inna) to zmień stan na q i przesuń głowicę do (następnej/poprzedniej) pozycji". Pokazać, że takie automaty rozpoznają wyłącznie języki regularne, oraz że niepustość jest też w PSPACE.*

Zadanie 6.6. *Wskazać algorytm wielomianowy, który sprawdza, czy dany automat alternujący dwukierunkowy akceptuje dane słowo wejściowe.*

Słowa nieskończone.

Zadanie 6.7. *Pokazać, że żaden nieskończony XOR nie jest językiem ω-regularnym.*

Zadanie 6.8. *Rozważmy grę G o pozycjach V, gdzie warunek wygrywający $W \subseteq V^\omega$ jest rozpoznawany przez automat $\mathcal{A}$ z warunkiem Büchiego o stanach Q. Rozważmy nową grę $G_\mathcal{A}$. Jej pozycje to $V \times Q$. Gra z pozycji (v,q) wygląda następująco. Najpierw gracz $\exists$ (niezależnie od tego, kto kontroluje v w grze G), wybiera stan p taki, że w automacie $\mathcal{A}$ jest przejście z q do p po literze $v \in V$. Następnie gracz który kontroluje v w grze G wybiera, zgodnie z zasadami G, nowy wierzchołek $w \in V$. Gra toczy się dalej z pary (w,p). Gracz $\exists$ wygrywa rozgrywkę, jeśli nieskończenie często na drugiej współrzędnej jest stan akceptujący.*

Pokazać, że gry G i $G_\mathcal{A}$ nie są równoważne. Dokładniej, pokazać, że może być pozycja v w grze G, w której wygrywa $\exists$, ale taka, że z pozycji (v,q_0) w grze $G_\mathcal{A}$ nie wygrywa $\exists$. (Gdzie q_0 to stan początkowy automatu $\mathcal{A}$.)

Zadanie 6.9. *Poprawić definicję gry $G_\mathcal{A}$ tak, aby była ona równoważna grze G, a jednocześnie była grą z warunkiem parzystości.*

Zadanie 6.10. *Wywnioskować z tego, że jeśli gracz ma strategię wygrywającą w grze G, to ma także wygrywającą strategię ze skończoną pamięcią.*

Zadanie 6.11. *Wywnioskować z tego, że automaty alternujące z warunkiem Mullera rozpoznają wyłącznie języki regularne.*

Zadanie 6.12. *Pokazać, że automaty alternujące dwukierunkowe rozpoznają wyłącznie języki regularne.*

Zadanie 6.13. *Rozważmy następujący problem* Parity*: dany skończony graf gry parzystości G (rozgrywki tej gry są nieskończone, pierwszy gracz wygrywa, gdy warunek parzystości jest spełniony) z wyróżnionym wierzchołkiem startowym x; pytamy, czy pierwszy gracz ma strategię wygrywającą.*

Pokazać, że problem Parity *jest w klasie NP$\cap$co-NP.*

Zadanie 6.14. *Rozwiązać problem* Parity *w czasie $O(n^d)$, gdzie d to liczba różnych rang występujących w G.* *Rozwiązanie na stronie 85.*

Zadanie 6.15. *Problem* RABINITY *powstaje przez zastąpienie w problemie* PARITY *warunku parzystości przez warunek Rabina. Rozgrywka spełnia warunek Rabina zadany przez ciąg zbiorów pozycji* $(F_1, Z_1), (F_2, Z_2), \ldots, (F_k, Z_k)$, *jeśli istnieje takie* i, *że pewna pozycja z* F_i *pojawia się nieskończenie często i każda pozycja z* Z_i *pojawia się tylko skończenie często. Pokazać, że problem* RABINITY *jest NP-zupełny.*

Zadanie 6.16. WEAKPARITY *powstaje przez zamianę warunku parzystości na słaby warunek parzystości: największy priorytet pojawiający się choć raz jest parzysty. Pokazać, że problem* WEAKPARITY *jest PTIME-zupełny (w sensie redukcji w LOGSPACE).* *Rozwiązanie na stronie 87.*

Zadanie 6.17. WEAKRABINITY *definiujemy podobnie, za pomocą słabego warunku Rabina: istnieje takie* i, *że jakaś pozycja z* F_i *się pojawia i żadna pozycja z* Z_i *się nie pojawia. Jaką złożoność ma ten problem?*

Rozdział 7

Drzewa

Spisali: Mateusz Kopeć, Maciek Zdanowicz i Adam Witkowski

7.1 Drzewa skończone

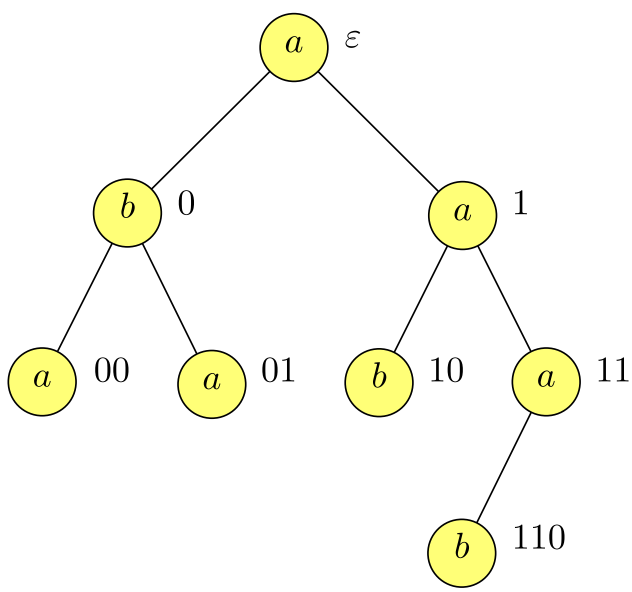
Będziemy rozważać drzewa binarne (każdy wierzchołek ma 0,1 lub 2 synów), uporządkowane (kolejność na synach jest określona, w szczególności wiemy, czy pojedynczy syn wierzchołka jest lewy czy prawy), nad alfabetem Σ. Możemy zdefiniować drzewo jako następującą funkcję, przypisującą wierzchołkom drzewa litery z alfabetu Σ:

$$t : D \to \Sigma$$

gdzie D opisuje wierzchołki drzewa binarnego, tak że korzeń opisany jest słowem ε, a synowie wierzchołka opisanego słowem w są opisani słowami $w \cdot 0$ (lewy syn) i $w \cdot 1$ (prawy syn). Zatem

$$D \subseteq \{0,1\}^* : \begin{cases} w \cdot 0 \in D \to w \in D \\ w \cdot 1 \in D \to w \in D \end{cases}$$

Przykład tak opisanego drzewa wygląda następująco:



Definicja 19. *Niedeterministyczny automat działający na drzewach to uporządkowana czwórka*

$$\mathcal{A} = (\Sigma, Q, \delta, Q_F)$$

gdzie

- Σ - *alfabet*

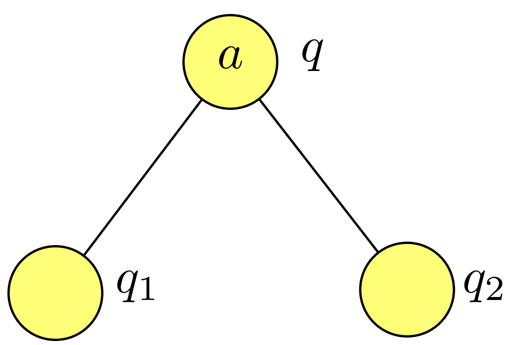
- Q - *zbiór stanów*

- $\delta \subseteq (Q \cup \{\cdot\}) \times (Q \cup \{\cdot\}) \times \Sigma \times Q$ - *relacja przejść*

- Q_F - *stany akceptujące w korzeniu*

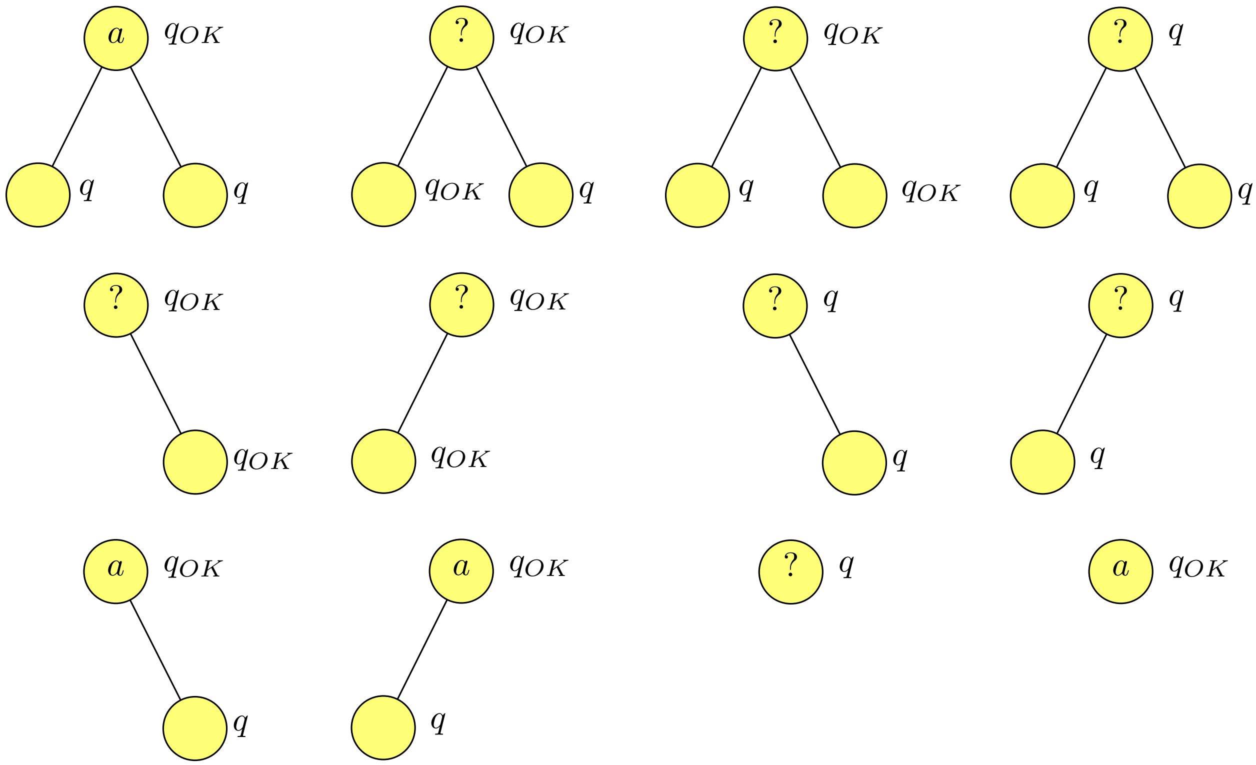
Relacja przejścia wyznacza dozwolone przypisania stanów wierzchołkom drzewa. Jeśli $(q_1, q_2, a, q) \in \delta$, dozwolone jest przypisanie stanu q do wierzchołka z literą a, o ile jego lewy syn ma przypisany stan q_1, a prawy q_2. Ponieważ wierzchołek może mieć tylko 1 syna lub nie mieć synów, dopuszczamy przejścia postaci $(\cdot, q_2, a, q)$, $(q_1, \cdot, a, q)$ i $(\cdot, \cdot, a, q)$.

Przykładowe przejście (q_1, q_2, a, q):



Bieg automatu to przypisanie stanów wszystkim wierzchołkom. Bieg akceptujący to bieg, w którym korzeniowi przypisano stan akceptujący.

Przykład. Automat $\mathcal{A}$ rozpoznający język L - „w drzewie istnieje litera a". Stany $\mathcal{A}$ to q_{OK} i q, stan akceptujący to q_{OK}. Przejścia wyglądają następująco: (? oznacza dowolną literę z alfabetu automatu)



Ten automat rozpoznaje L, bo po napotkaniu litery a przypisuje stan q_{OK}, który następnie jest propagowany w górę, do korzenia. $\square$

Rozważmy teraz deterministyczną wersję automatów na drzewach. Relacja przejścia jest zastąpiona przez funkcję przejścia. Możliwe są dwa podejścia - przypisywanie stanów idąc w dół lub w górę drzewa.

Definicja 20. *Deterministyczny automat działający na drzewach (wersja z dołu do góry) to uporządkowana piątka*

$$\mathcal{A} = (\Sigma, Q, \delta, Q_F, q_I)$$

gdzie

- Σ - *alfabet*
- Q - *zbiór stanów*
- $\delta : (Q \cup \{\cdot\}) \times (Q \cup \{\cdot\}) \times \Sigma \to Q$ - *funkcja przejść*
- Q_F - *stany akceptujące w korzeniu*
- q_I - *stan startowy, przyjmujemy go wszędzie tam, gdzie nie ma syna*

Automat taki przypisuje stany kolejnym wierzchołkom zgodnie z funkcją przejść, poczynając od liści i stopniowo przesuwając się w stronę korzenia. Warunkiem akceptacji drzewa jest przypisanie korzeniowi stanu akceptującego (należącego do Q_F).

Definicja 21. *Deterministyczny automat działający na drzewach (wersja z góry do dołu) to uporządkowana piątka*

$$\mathcal{A} = (\Sigma, Q, \delta, Q_F, q_I)$$

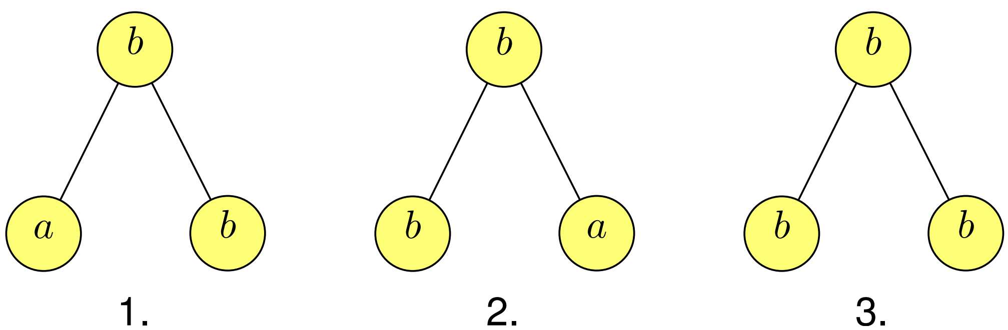
gdzie

- Σ - *alfabet*
- Q - *zbiór stanów*
- $\delta : (Q \cup \{\cdot\}) \times \Sigma \to Q \times Q$ - *funkcja przejść*
- Q_F - *stany akceptujące, sprawdzane wszędzie tam, gdzie nie ma syna*
- q_I - *stan startowy w korzeniu*

Ta wersja automatu przypisuje stany kolejnym wierzchołkom zgodnie z funkcją przejść, poczynając od korzenia i stopniowo schodząc w dół w stronę liści drzewa. Warunkiem akceptacji drzewa jest przypisanie stanów należących do Q_F wszędzie tam, gdzie nie ma syna.

Najpierw zajmijmy się wersją z góry do dołu.

Przykład. Niech $L = \{t : \text{któryś wierzchołek to } a\}$. Przypuśćmy, że istnieje automat deterministyczny w wersji z góry do dołu, rozpoznający L. Rozważmy trzy słowa:



Automat, zaczynając w stanie q_I, po przeczytaniu w korzeniu litery b, zgodnie ze swoją funkcją przejść δ przypisuje niżej stany q_1 i q_2. Obydwa stany muszą być akceptujące, aby automat zaakceptował słowo 1. i słowo 2., należące do L. Automat zaakceptuje też zatem słowo 3., które nie należy do L.

L nie da się zatem rozpoznać za pomocą automatu w wersji z góry do dołu. □

Wniosek 34. *Automaty deterministyczne w wersji działającej z góry do dołu rozpoznają istotnie mniej niż automaty niedeterministyczne.*

Twierdzenie 35. *Dowolny automat niedeterministyczny $\mathcal{A}$ ze stanami Q można zdeterminizować, to znaczy istnieje automat deterministyczny w wersji z dołu do góry rozpoznający $L(\mathcal{A})$.*

Dowód
Pokażemy, że automat $\hat{\mathcal{A}} = (\Sigma, \hat{Q}, \hat{\delta}, \hat{Q_F}, \hat{Q_I})$ determinizuje automat $\mathcal{A} = (\Sigma, Q, \delta, Q_F)$, czyli $L(\mathcal{A}) = L(\hat{\mathcal{A}})$. Automat $\hat{\mathcal{A}}$ zdefiniujmy nastepująco:

$$\hat{Q} = P(Q)$$

$$\hat{Q_I} = \{Q_I\}$$

$$\hat{Q_F} = \{\hat{q_F} : \hat{q_F} \cap Q \neq \emptyset\}$$

$$\hat{\delta} = \{(Q_1, Q_2, a, Q_3) : Q_3 = \{q : \exists q_1 \in Q_1, q_2 \in Q_2 (q_1, q_2, a, q) \in \delta\}\}$$

Tu by się przydała jakaś agitacja, czemu to działa.

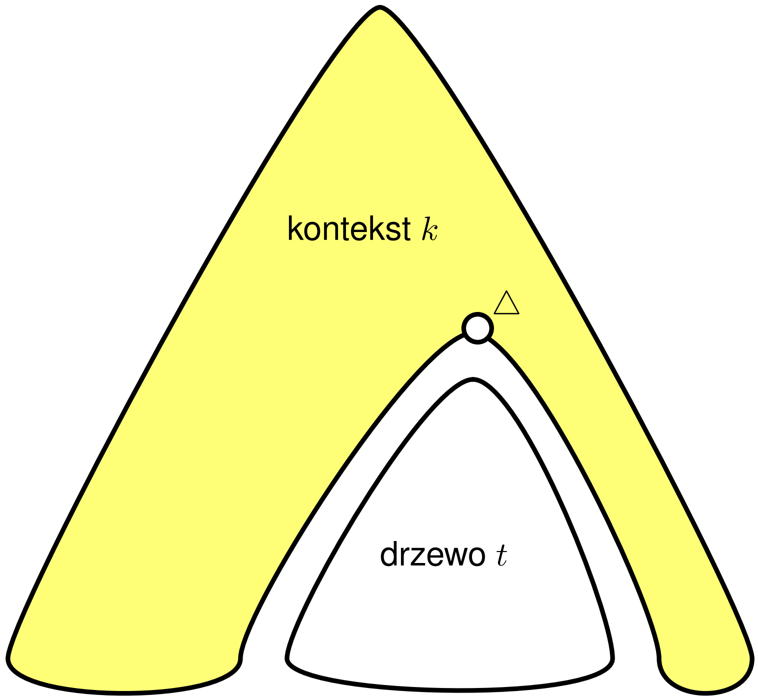
Jeśli mamy akceptujący bieg automatu niedeterministycznego, to indukcja w górę. W drugą stronę - odtwarzamy od góry bieg automatu niedeterministycznego. □

Zastanówmy się teraz, jak wygląda minimalny (dla danego języka) deterministyczny automat działający na drzewach.

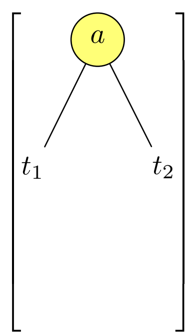
Ustalmy język L i wprowadźmy relację równoważności $\sim_L$:

$$t_1 \sim_L t_2 \iff \forall k (k \cdot t_1 \in L \iff k \cdot t_2 \in L)$$

gdzie k to kontekst, czyli drzewo z dokładnie jednym wyróżnionym wierzchołkiem - liściem o etykiecie $\triangle$, natomiast $k \cdot t$ to drzewo powstałe z kontekstu k poprzez zastąpienie $\triangle$ przez drzewo t.



Definicja 22. *Minimalny automat na drzewach dla języka L definiujemy następująco:*

- *Stany automatu minimalnego to klasy abstrakcji relacji $\sim_L$.*
- *Stan początkowy: $[drzewopuste]$.*
- *Stany akceptujące: $\{[t] : t \in L\}$.*
- *Przejścia są zdefiniowane następująco:* $\delta([t_1],[t_2],a) =$  .

Zauważmy, że jeśli drzewa t_1 i t_2 po zastosowaniu do niech automatu uzyskują w korzeniu ten sam stan q, to $t_1 \sim_L t_2$. Minimalny automat $\mathcal{A}$ jest „lepszy” od każdego innego deterministycznego automatu rozpoznającego L (to znaczy posiada najmniejszą ilość stanów).

Przejdźmy teraz do związku automatów z MSO ze zdefiniowanymi predykatami $\swarrow$ i $\searrow$, oznaczającymi odpowiednio lewego i prawego syna danego wierzchołka.

Twierdzenie 36. *$MSO(\swarrow, \searrow)$ wyraża to samo, co automaty.*

Dowód ($\leftarrow$)
Formuła opisująca ten sam język co automat wygląda następująco:

$$\exists X_1 \exists X_2 \ldots \exists X_n (\text{każdy wierzchołek należy do jakiegoś ze zbiorów} \wedge \text{przejścia})$$

X_1 to zbiór wierzchołków, którym automat w biegu akceptującym przypisał pierwszy stan, X_2 - drugi stan, itd... □

Dowód ($\rightarrow$)
Dowód opiera się na następujących faktach:

1. proste języki da się rozpoznawać
2. negacja - bo istnieją automaty deterministyczne - możemy zamienić stany akceptujące na nieakceptujące i odwrotnie
3. suma - łatwe (w niedet)
4. rzutowanie - łatwe (w niedet)

□

7.2 Drzewa nieskończone

Naturalne wydaje się rozszerzenie naszych dotychczasowych rozważań o automaty działające na drzewach nieskończonych. Oczywiście będą interesować nas teraz wyłącznie automaty niedeterministyczne, ponieważ automaty deterministyczne w wersji z dołu do góry nie mają skąd zacząć przetwarzać drzewo (bo jest nieskończone), a w wersji z góry do dołu nigdy nie skończą rzeczonego przetwarzania (bo drzewo jest nieskończone).

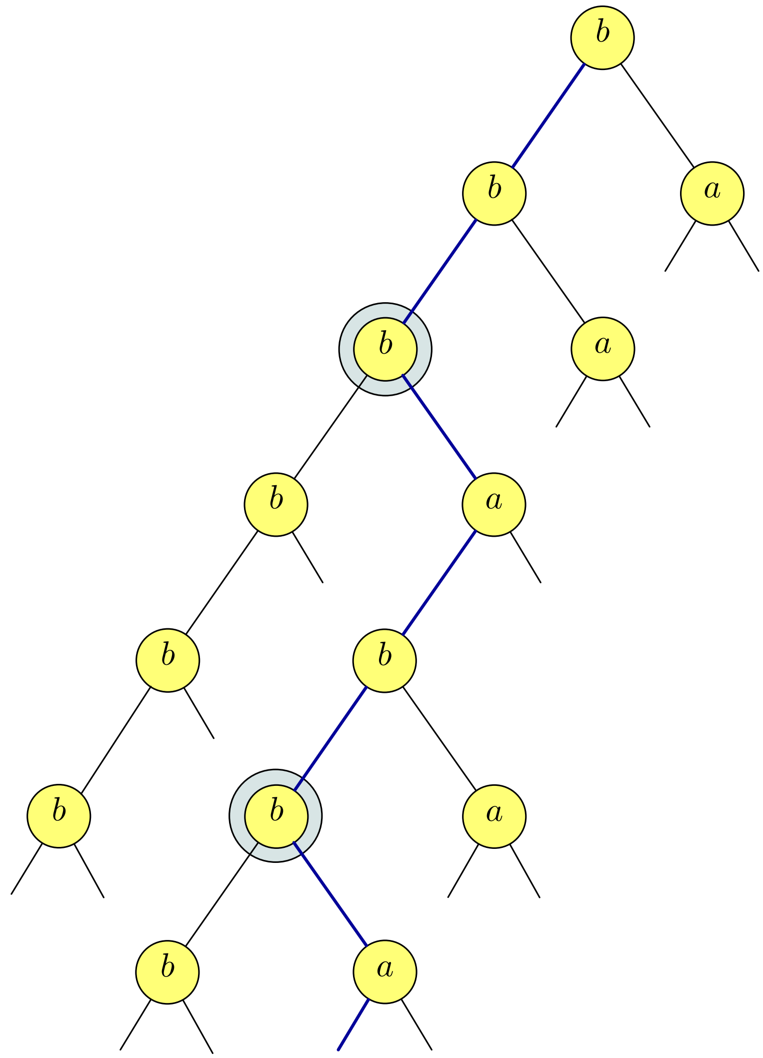
Drzewo nieskończone jest binarnym drzewem pełnym, to znaczy każdy wierzchołek ma dokładnie dwóch synów. Ulega zmianie akceptacja automatu - teraz automat akceptuje dane słowo $\iff$ na każdej ścieżce spełniony jest dany warunek akceptacji. Warunkiem akceptacji mogą być znane nam warunki: Büchiego, Müllera, parzystości.

Przykład. Zdefiniujmy trzy języki:
L_1 = na pewnej ścieżce nieskończenie wiele a.
L_2 = na każdej ścieżce skończenie wiele a.
L_3 = na każdej ścieżce słowo z języka L.

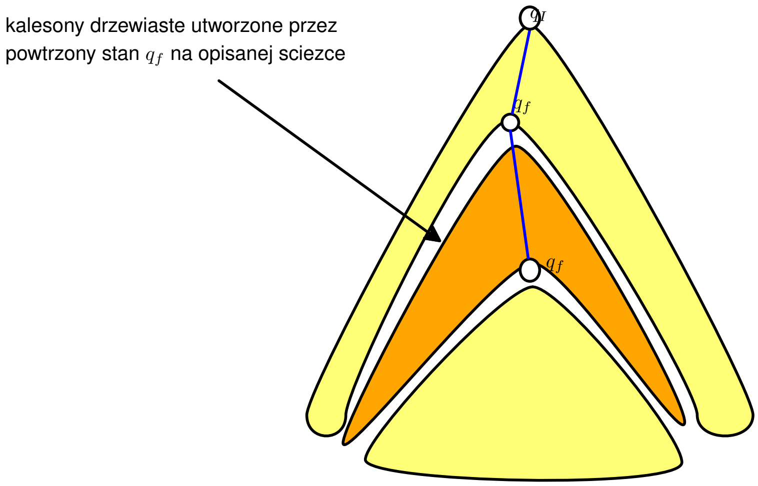
L_1 można rozpoznawać automatem z warunkiem Müllera.
L_2 można rozpoznawać automatem z warunkiem Müllera.
L_3 można rozpoznawać automatem z warunkiem Müllera, o ile język L jest rozpoznawany przez automat deterministyczny. □

Twierdzenie 37. *Automaty drzew nieskończonych z warunkiem Büchiego rozpoznają istotnie mniej niż automaty z warunkiem Müllera.*

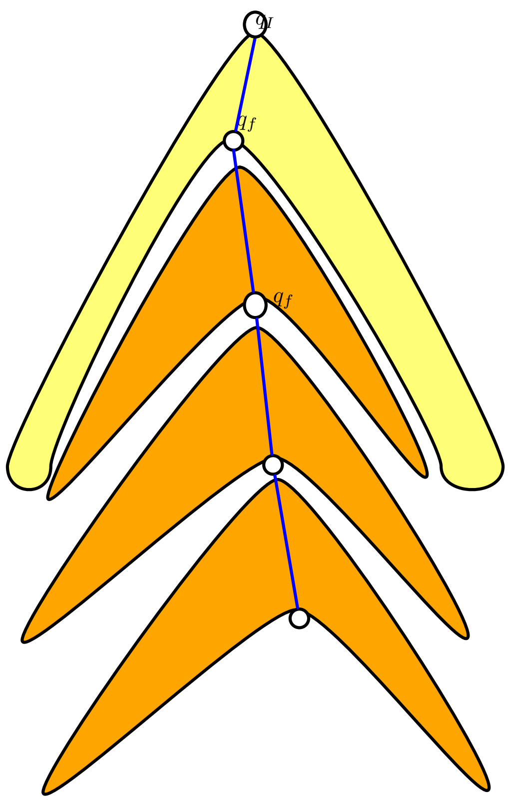
Dowód
Rozważmy na przykład język L_2. Przypuśćmy, że automat $\mathcal{A}$ z warunkiem Büchiego rozpoznaje L_2. $\mathcal{A}$ ma n stanów.
Rozważmy następujące słowo S z języka L_2:
wierzchołkom $(0^*1)^k$ dla $k \leq n+1$ przypisujemy a, pozostałym wierzchołkom b.



Ponieważ S należy do L_2 na ścieżce 0^* musi znajdować się jakiś wierzchołek v_1 oznaczony stanem akceptującym q_{f_1}. Rozważając prawego syna wierzchołka v_1 i wiodącą z niego maksymalnie lewą ścieżkę, dostajemy wierzchołek v_2 ze stanem akceptującym q_{f_2} (Rys. powyżej - oznaczone błękitnym kolorem).
Powtarzając tę procedurę (schodzenie do prawego syna z akceptującego wierzchołka i rozważanie ścieżki najbardziej w lewo) $n+1$ razy dostajemy ciąg par (wierzchołek,stan akceptujący): $(v_1,q_{f_1}),(v_2,q_{f_2}),..,(v_{n+1},q_{f_{n+1}})$. Ze względu na ilość stanów automatu $\mathcal{A}$, któryś stan q_{f_i} musi się powtórzyć, tworząc zaprezentowane poniżej kalesony.



Zamieniając kontynuację słowa zaczepioną w dolnym wierzchołku q_f przez wielokrotnie połączone kalesony, dostajemy słowo, które będzie spełniać warunek Büchiego na każdej ścieżce (jedyna wątpliwa ścieżka zawiera nieskończenie wiele stanów q_f), ale nię będzie należeć do języka L_2 (dołączenie jednych kalesonów zwiększa liczbę liter a na ścieżce o 1).



To daje nam sprzeczność z przypuszczeniem, że automat $\mathcal{A}$ rozpoznawał język L_2. Jak już wspomnieliśmy, język ten jest rozpoznawany przez automat z warunkiem Müllera, co daje nam tezę twierdzenia.
□

Formalnie, drzewem nieskończonym t będziemy nazywać funkcję $t : \{0,1\}^* \to A$, gdzie A to skończony alfabet. Możemy rozpatrywać takie drzewo jako strukturę logiczną:

$$\underline{t} =< \{0,1\}^*, \{a(x)\}_{a\in A}, s_0(x,y), s_1(x,y) > \tag{7.1}$$

gdzie $a(x)$ jest prawdą $\iff$ wierzchołek x ma przypisaną literę a, $s_0(x,y)$ znaczy x jest lewym synem y a $s_1(x,y)$ oznacza x jest prawym synem y. Zauważmy, że do struktury moglibyśmy dodać porządek starszeństwa, ale nie jest to konieczne, gdyż taki porządek można łatwo zdefiniować.

Twierdzenie 38. *Spełnialność formuł logiki monadycznej jest rozstrzygalna w drzewach nieskończonych, tzn. rozstrzygalny jest nastepujący problem:*
Dana jest formuła $\phi \in MSO(s_0, s_1)$ nad alfabetem A.
Pytanie: Czy ϕ jest prawdziwa w pewnym drzewie t?.

To twierdzenie zostało udowodnione w 1969 roku przez Rabina w sformułowaniu:

Twierdzenie 39. *Teoria MSO struktury $R =< \{0,1\}^*, s_0(x,y), s_1(x,y) >$ jest rozstrzygalna.*

Strukturę R nazywamy drzewem Rabina.

Zanim przystąpimy do właściwego dowodu, nakreślmy jego ideę:
Najpierw zdefiniujemy automaty alternujące i niedeterministyczne dla drzew nieskończonych i udowodnimy równoważność tych dwóch klas automatów. Następnie pokażemy, że MSO na drzewach nieskończonych jest równoważna automatom na takich drzewach. Na koniec udowodnimy, że problem pustości dla automatów na drzewach jest rozstrzygalny.

Definicja 23. *Warunek parzystości dla ścieżki w drzewie nieskończonym.*
Zakładamy, że każdy wierzchołek na ścieżce ma przypisaną rangę, czyli liczbę naturalną ze zbioru $\{0, \cdots, n\}$. Warunek parzystości dla ścieżki jest spelniony, gdy największa z rang, które wystąpiły nieskończenie często, jest liczbą parzystą.

Mając zdefiniowany warunek parzystości dla ścieżki w drzeenie nieskończonym, możemy przystąpić do definicji automatów niedeterministycznych i alternujących. Oba automaty na wejściu dostają drzewo nieskończone t.

Definicja 24. *Automat niedeterministyczny z warunkiem parzystości dla drzew nieskończonych.*
Automat niedeterministyczny $\mathcal{A}$ to piątka: $< Q, q_\epsilon, A, \delta, \Omega >$, gdzie

Q *- stany automatu*

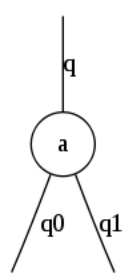
q_ϵ *- stan początkowy*

A *- alfabet*

$\delta \subset Q \times A \times Q \times Q$ *- przejścia*

$\Omega : Q \to \{0, \cdots, n\}$ *- rangi*

Bieg automatu to przyporządkowanie stanów wierzchołkom, czyli funkcja $\rho : \{0,1\}^* \to Q$. Przejścia opisują możliwe przypisania stanów, tzn. przejście $(q, a, q0, q1) \in \delta$ oznacza, że jeśli wierzchołkowi drzewa v o etykiecie a jest przypisany stan q, to jego lewemu synowi można przypisać $q0$, jednocześnie przypisując prawemu synowi vstan $q1$. Przykładowe przejście:



Zakładamy, że korzeń ma przypisany stan q_ϵ Bieg ρ nazywamy akceptującym, gdy bieg jest zgodny z δ, w korzeniu jest q_ϵ oraz na każdej ścieżce spełniony jest warunek parzystości z definicji 23. Automat $\mathcal{A}$ akceptuje drzewo, gdy dla tego drzewa $\mathcal{A}$ ma bieg akceptujący.

Definicja 25. *Automat alternujący z warunkiem parzystośc dla drzew nieskończonych.*
Automat alternujący $\mathcal{A}$ to szóstka: $< Q_\forall, Q_\exists, q_\epsilon, A, \delta, \Omega >$, gdzie

> $Q_\forall$ *- stany gracza "dla każdego"*
>
> $Q_\exists$ *- stany gracza "istnieje"*
>
> q_ϵ *- stan początkowy*
>
> A *- alfabet*
>
> $\delta \subset Q \times A \times Q \times \{\epsilon, 0, 1\}$ *- przejścia*
>
> $\Omega : Q \to \{0, \cdots, n\}$ *- rangi*

$Q = Q_\forall \cup Q_\exists$ *i* $Q_\forall \cap Q_\exists = \emptyset$. *Przejście* (q,a,p,x) *oznacza, że automat będąc w stanie* q*, widząc wierzchołek* v *o etykiecie* a *może przejść do stanu* p*, jednocześnie obserwowanym wierzchołkiem zostaje:* $\begin{cases} \textit{lewy syn } v & x = 0 \\ v & x = \epsilon \\ \textit{prawy syn } v & x = 1 \end{cases}$

Aby określić warunek akceptacji dla $\mathcal{A}$, zdefiniujmy grę $G_t^{\mathcal{A}}$ (t to drzewo, które automat otrzymał na wejściu).
Nasz automat będzie akceptował drzewo t, gdy gracz $\exists$ ma strategię wygrywającą dla $G_t^{\mathcal{A}}$
Stany gry to pary $(q, v) \in Q \times \{0,1\}^*$. Gra zaczyna się w (q_ϵ, ϵ). Jest dwóch graczy, $\exists$ i $\forall$. W stanie (q, v) ruch wykonuje ten gracz, do którego należy q, tzn. jeśli $q \in Q_\exists$ to ruch wykonuje gracz $\exists$, itd. Przejścia w grze muszą być zgodne z δ, czyli przejściami automatu. Przebiegiem gry $G_t^{\mathcal{A}}$ nazwiemy ciąg stanów gry ρ, zaczynający się w (q_ϵ, ϵ), taki, że pary kolejnych wyrazów ciągu (tzn. pary $(q_i, v_i), (q_{i+1}, v_{i+1})$) to poprawne przejścia w grze $G_t^{\mathcal{A}}$. Zauważmy, że taki ciąg opisuje ścieżkę w drzewie t, ponadto do każdego wierzchołka tej ścieżki jest przypisany stan automatu. Gracz $\exists$ wygrywa grę $G_t^{\mathcal{A}}$ o przebiegu ρ gdy jest spełniony warunek parzystości dla ścieżki określonej przez ρ.

Do końca tego wykładu wszystkie automaty będą akceptować drzewa zgodnie z warunkiem parzystości.

Lemat 40. *Dla drzew nieskończonych następujące klasy automatów są równoważne:*

1. *automaty niedeterministyczne.*
2. *automaty alternujące.*

Dowód $(1 \to 2)$
Rozważmy dowolny automat niedeterministyczny $\mathcal{A} = \langle Q_\mathcal{A}, q_\epsilon, A, \delta_\mathcal{A}, \Omega \rangle$. Skonstruujemy równoważny automat alternujący $\mathcal{B}$. Stany $\mathcal{B}$ zdefinujemy następująco: $Q_\mathcal{B} = Q_\mathcal{A} \cup \delta_\mathcal{A}$. Oryginalne stany automatu $\mathcal{A}$ zostaną stanami $\exists$, a stany-przejścia będą stanami $\forall$. Stan początkowy automatu się nie zmienia. Rangi oryginalnych stanów $\mathcal{A}$ pozostają bez zmian, za to stany-przejścia otrzymują rangi 0. Gdy $\mathcal{B}$ widzi wierzchołek v z literą a, możliwe są następujące przejścia:

- ze stanu $q \in Q_\mathcal{A}$ do każdego $(q, a, q_0, q_1) \in \delta_\mathcal{A}$. Przy takim przejściu automat pozostaje w tym samym wierzchołku drzewa.
- ze stanu będącego przejściem $(q,a, q_0, q_1) \in \delta_\mathcal{A}$ do q_0, jednocześnie schodząc w lewo w drzewie lub q_1, przy jednoczesnym zejściu w prawo.

Intuicyjnie, podczas biegu $\mathcal{B}$, gdy $\mathcal{B}$ jest w stanie $q \in Q_\mathcal{A}$ gracz $\exists$ wybiera z możliwych przejść $\mathcal{A}$ jedno, a następnie gracz $\forall$ wybiera czy za pomocą takiego przejścia pójść w prawo czy w lewo.

Zauważmy, że każda strategia gracza $\exists$ wyznacza pewien bieg automatu $\mathcal{A}$ (i każdy dozwolony bieg jest przez pewną strategię wyznaczany) - strategia dla drzewa t mówi jakiego przejścia użyć w wierzchołku v,

który ma stan q i literę a, co wystarcza do określenia które stany przypisać poszczególnym wierzchołkom, jeśli założymy, że korzeniowi przypisujemy q_ϵ.

Z kolei strategia gracza $\forall$ polega na wybraniu ścieżki w drzewie, starając się, by na tej ścieżce nie został spełniony warunek parzystości.

Jeśli w automacie $\mathcal{A}$ jest bieg akceptujący dla drzewa t, to dowolna strategia gracza $\exists$ wyznaczająca ten bieg jest wygrywająca, gdyż wtedy na każdej ścieżce w t spełniony jest warunek parzystości.

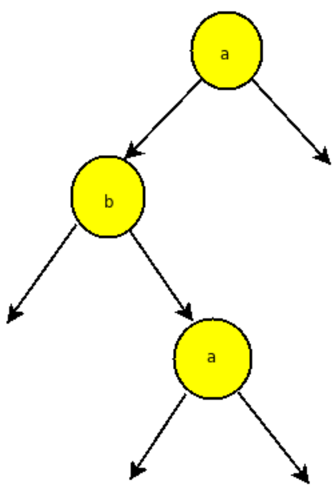
Musimy jeszcze pokazać, że gdy gracz $\exists$ nie ma strategii wygrywającej dla t, to $\mathcal{A}$ nie akceptuje t. Jeśli $\exists$ nie ma strategii wygrywającej, to taką strategię ma $\forall$. Wtedy, dla każdej strategii $\exists$, gracz $\forall$ może tak wybierać ścieżkę w drzewie, że na tej ścieżce nie będzie spełniony warunek parzystości. Z faktu, że każdy bieg $\mathcal{A}$ na drzewie t odpowiada jakiejś strategii gracza $\exists$, otrzymujemy, że istnienie strategii wygrywającej dla gracza $\forall$ dla drzewa t implikuje to, że $\mathcal{A}$ nie akceptuje t. To kończy dowód. □

Dowód ($2 \rightarrow 1$)

Do dowodu tej równoważności będziemy potrzebować następującego lematu:

Lemat 41 (o każdej ścieżce)**.** *Niech A - pewien skończony alfabet. Niech $K \subset (A \times \{0{,}1\})^\omega$ - język ω-regularny. Wówczas język ńa każdej ścieżce K"jest rozpoznawany przez automat niedeterministyczny dla drzew.*

Dowód
Niech A, K jak w tezie lematu. Na początek zauważmy, że każde słowo $v \in K$ opisuje ścieżkę w pewnym drzewie nieskończonym t.
Np. $v = (a{,}0)(b{,}1)(a{,}1)$... odpowiada następującej ścieżce :



Pominięto tutaj wszystkie pozostałe wierzchołki drzewa.

Niech $\mathcal{A}$ - automat deterministyczny rozpoznający K. Konstruujemy automat niedeterministyczny $\mathcal{A}'$, taki że $\mathcal{A}'$ akceptuje drzewo $t \iff$ każda ścieżka t rozumiana jako słowo nad $(A \times \{0{,}1\})^\omega$ należy do K. W $\mathcal{A}'$ stany (ozn. Q), stan początkowy, alfabet i rangi pozostają jak w automacie $\mathcal{A}$. Przejścia $\mathcal{A}'$ konstruujemy następująco:
Dla każdego stanu $q \in Q$, dla każdej litery $a \in A$ dodajemy przejście $(q, a, q_0, q_1) \in Q \times A \times Q \times Q$, takie, że q_0 to stan $\mathcal{A}$ do którego $\mathcal{A}$ przechodzie z stanu q po przeczytaniu $(a{,}0)$, a q_1 to stan $\mathcal{A}$ do którego $\mathcal{A}$ przechodzie z stanu q po przeczytaniu $(a{,}1)$.

Oczywiście $\mathcal{A}'$ akceptuje drzewo t tylko jeśli każda ścieżka drzewa (rozumiana jako słowo) należy do K. □

Rozważmy dowolny automat alternujący $\mathcal{A}$. Taki automat akceptuje drzewo $t : \{0{,}1\}^* \rightarrow A$ gdy gracz $\exists$ ma

strategię wygrywającą w $G_t^{\mathcal{A}}$. Pozycje w grze to pary $(q, v) \in Q \times \{0,1\}^*$. Widać, że strategia $\sigma_\exists$ to drzewo $\sigma_\exists : \{0,1\}^* \to (Q \times \{0, 1, \epsilon\})^{Q_\exists}$ (to znaczy, że alfabetem są funkcje z $Q_\exists w Q \times \{0, 1, \epsilon\}$) Rozważmy

$$L = \{t \otimes \sigma_\exists | \sigma_\exists \text{ jest strategią wygrywającą w } G_t^{\mathcal{A}}\} \tag{7.2}$$

Język $\mathcal{A}$ to rzut L na alfabet A. Wystarczy pokazać, że L jest rozpoznawany przez automat niedeterministyczny. L jest językiem ńa kazdej ścieżce K", gdzie K to "każdy wybór strategii dla $\forall$ który pozostaje na ścieżce kończy się zwycięstwem $\exists$". Oczywiscie jest to język ω-regularny. Stąd, z lematu o każdej ścieżce wynika, że żądany automat niedeterministyczny istnieje. □

Ostatnim elementem dowodu rozstrzygalności MSO na drzewach nieskończonych jest pokazanie, że problem pustości języka automatu niedeterministycznego jest rozstrzygalny.

Twierdzenie 42. *Problem niepustości:*

Dane: Automat niedeterministyczny z warunkiem parzystości dla drzew nieskończonych $\mathcal{A}$.
Pytanie: Czy istnieje takie drzewo t, że $\mathcal{A}$ akceptuje t?

jest rozstrzygalny.

Dowód
Mając dany $\mathcal{A} = \langle Q, q_\epsilon, A, \delta, \Omega \rangle$ konstruujemy grę parzystości o skończonej planszy $H_{\mathcal{A}}$, taką, że jeden z graczy (gracz Automat) ma strategię wygrywającą w tej grze $\iff$ istnieje drzewo t, które $\mathcal{A}$ akceptuje. Jak wiadomo, rozwiązywanie gier parzystości o skończonej planszy jest w $\mathbb{NP} \wedge co - \mathbb{NP}$, więc jeśli potrafimy taką grę skonstruuować, to problem niepustości jest rozstrzygalny.

Przejdźmy do konstrukcji $H_{\mathcal{A}}$. W grze będzie dwóch graczy, Automat i Tropiciel. Stany gry $Q_{gry} = Q \cup \delta$, tzn. stanami gry są stany i przejścia automatu $\mathcal{A}$. Rangi stanów gry będących stanami $\mathcal{A}$ zachowują rangi z $\mathcal{A}$, stany gry będące przejściami $\mathcal{A}$ otrzymują rangi 0. Zauważmy, że rangi stanów-przejść nie mają wpływu na wynik gry (jeżeli 0 jest najwi ększą rangą, która wystąpiła nieskończenie często podczas gry, to 0 jest jedyną rangą, która wystąpiła nieskończenie często).

Przebieg gry wygląda następująco: Stanem początkowym $H_{\mathcal{A}}$ jest q_ϵ - stan początkowy $\mathcal{A}$. W stanach gry będących stanami $\mathcal{A}$ rusza się gracz Automat - wybiera jedno przejście (w ten sposób wybiera też literę), które jest możliwe z obecnego stanu w $\mathcal{A}$. W stanie będącym przejściem $\mathcal{A}$, gracz Tropiciel wybiera, do którego z dwóch możliwych stanów przechodzimy.

Automat wygrywa, gdy warunek parzystości jest spełniony, tzn. największa ranga spośród rang występujących nieskończenie często podczas gry jest parzysta.

Intuicyjnie, gra polega na tym, że Automat wybiera, jakie jest drzewo, które $\mathcal{A}$ mógłby zaakceptować, jednocześnie proponując przypisanie wierzchołkom stanów $\mathcal{A}$ a Tropiciel stara się znaleźć najgorszą możliwą ścieżkę.

Jeśli istnieje takie drzewo, które $\mathcal{A}$ akceptuje, to istnieje dla tego drzewa bieg akceptujący, tzn. takie przypisanie stanów wierzchołkom drzewa, że na każdej ścieżce jest spełniony warunek parzystości. Wtedy oczywiście jeśli Automat wybiera przejścia zgodne z takim biegiem, to niezależnie od wyborów Tropiciela, warunek parzystości będzie spełniony, a więc Automat ma strategię wygrywającą w $H_{\mathcal{A}}$.

Z kolei gdy nie ma drzewa, które $\mathcal{A}$ akceptuje, to dla każdego drzewa, dla każdego biegu automatu istnieje taka ścieżka w drzewie z przypisanymi stanami, na której warunek parzystości nie jest spełniony. Wystarczy, że Tropiciel będzie wybierał kierunki przejścia zgodne z taką ścieżką, wtedy Automat nie może wygrać, więc nie ma strategii wygrywającej. □

Wiedząc to wszystko, możemy przystąpić do dowodu głównego twierdzenia tego wykładu, dla przypomnienia sformułujmy je raz jeszcze:

Twierdzenie 43. *Spełnialność formuł logiki monadycznej jest rozstrzygalna w drzewach nieskończonych, tzn. rozstrzygalny jest nastepujący problem:*
Dana jest formuła $\phi \in MSO(s_0, s_1)$ nad alfabetem A.
Pytanie: Czy ϕ jest prawdziwa w pewnym drzewie t?

Dowód
Jasne jest, że dla prostych formuł można skonstruować równoważne im automaty dla drzewach. Jak wiadomo automaty niedeterministyczne są zamknięte an operację rzutowania, z kolei automaty alternujące są zamknięte na negację. Ponieważ dla drzew nieskończonych te klasy automatów są równoważne i obie te klasy są zamknięte ze względu na sumę to, z poprzednio udowodnionych twierdzeń wiemy, że formuły MSO są równoważne automatom.

Z twierdzenia 42 wiemy, że problem pustości jest rozstrzygalny dla automatów niedetereministycznych, a więc także problem spełnialności dla formuł logiki MSO jest rozstrzygalny. □

Zadania

Drzewa skończone.

Zadanie 7.1. *Pokazać, że dla automatów niedeterministycznych i deterministycznych (top-down) problem pustości jest PTIME-zupełny (w sensie redukcji w LOGSPACE).*

Zadanie 7.2. *Pokazać, że dla automatów niedeterministycznych problem uniwersalności (tj. sprawdzenie, czy automat akceptuje każde drzewo) jest EXPTIME-zupełny.*

Zadanie 7.3. *Problem niepustości przecięcia polega na rozstrzygnięciu dla danego* n *i danych automatów* $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$, *czy* $L(\mathcal{A}_1) \cap L(\mathcal{A}_2) \cap \ldots \cap L(\mathcal{A}_n) \neq \emptyset$. *Pokazać, że (dla deterministycznych i niedeterministycznych automatów) problem niepustości przecięcia jest EXPTIME-zupełny.*

Drzewa nieskończone.

Zadanie 7.4. *Czy język „nieskończenie wiele wierzchołków ma etykietę* a*" jest regularny?*

Zadanie 7.5. *Czy dla każdej formuły WMSO istnieje równoważna formuła MSO?*

Rozdział 8

Inne teorie rozstrzygalne

Spisali: Michał Gołębiowski

Przypomnijmy, że na mocy poprzedniego rozdziału wiemy, że zachodzi

Twierdzenie 44. *Teoria MSO drzewa Rabina*

$$\langle \{0,1\}^*, s_0(x,y), s_1(x,y) \rangle$$

jest rozstrzygalna.

W rozdziale tym powrócimy do interpretacji zdefiniowanych w rozdziale 3 i spróbujemy pokazać parę konsekwencji powyższego twierdzenia.

Twierdzenie 45. *Teoria FO struktury $\langle \mathbb{N}, \cdot \rangle$ jest rozstrzygalna.*

Dowód. Jako że interpretacje były zdefiniowane dla struktur czysto relacyjnych, traktujemy tu mnożenie jako relację trzyargumentową, do której trójka (x, y, z) należy wtedy i tylko wtedy, gdy $x \cdot y = z$. Będziemy jednak stosować zwyczajowe oznaczenia.

Ponieważ zajmujemy się logiką FO, adekwatną będzie interpretacja element-zbiór, tj. liczby będziemy interpretować jako zbiory w drzewie Rabina. Przedstawiamy mianowicie liczbę n jako iloczyn $n = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\alpha_k}$, gdzie p_i to i-ta liczba pierwsza, zaś $\alpha_k \neq 0$ (jest to oczywiście przedstawienie jednoznaczne). Każdy z czynników $p_i^{\alpha_i}$ interpretujemy osobno. Liczbę $p_i^{\alpha_i}$ będziemy reprezentować jako podzbiór gałęzi $0^i 1^*$, gdzie na części składającej się z jedynek kodujemy binarnie w porządku odwrotnym (tj. najmniej znaczące bity są najbliżej „zakrętu") liczbę α_i.

Zdefiniujmy formułę $\varphi_{\mathrm{el}}(X)$ (patrz: rozdział 3), która będzie wyrażać zdanie: „X jest skończonym podzbiorem 0^*1^*" jako koniunkcję formuł $\alpha(X)$ i $\beta(X)$, gdzie:

(a) $\alpha(X) \;\equiv\;$ X jest skończony

(b) $\beta(X) \;\equiv\; X \subseteq 0^*1^*$

Aby zdefiniować $\beta(X)$, zauważmy, że musimy zapisać własność: „jeśli w drodze od korzenia do wierzchołka $x \in X$ skręcimy na drzewie w prawo, to potem nie skręcimy już nigdy w lewo", a więc:

$$\beta(X) \;\equiv\; \forall_{x \in X} \neg \exists_{y_1, y_2, y_3 \leq x} \; s_1(y_1, y_2) \wedge s_0(y_2, y_3)$$

Pozostało zdefiniować drugą z formuł. W tym celu zauważmy, że zbiór X jest nieskończony wtedy i tylko wtedy, gdy zbiór prefiksy(X) jest nieskończony. Zbiór prefiksy(X) jest zamknięty na obcięcia, a stąd oraz z faktu, że drzewo Rabina ma skończone rozgałęzienie na mocy lematu Königa wynika, że zbiór prefiksy(X) ma nieskończoną ścieżkę tworzącą gałąź w drzewie Rabina. Stąd mamy:

$$\alpha(X) \;\equiv\; \exists_Y \left((\forall_{y \in Y} \; \exists_{x \in X} \; y \leq x) \wedge (\forall_{y \in Y} \; \exists_{z \in Y} \; s_0(y,z) \vee s_1(y,z)) \right)$$

Pierwsza z podformuł głównej koniunkcji w $\alpha(X)$ mówi, że Y zawiera się w zbiorze prefiksy(X), zaś druga, że każdy element Y ma następnika w Y (czyli że Y zawiera nieskończoną gałąź). Pozostaje zdefiniować interpretację relacji mnożenia w tej strukturze, tj. formułę $\varphi_\bullet(X,Y,Z)$. W tym celu zdefiniujmy najpierw taką formułę przy założeniu, że X, Y i Z definiują liczby będące potęgami dwójki, a zatem leżą w całości na gałęzi 1^* (oznaczmy ją przez $\psi(X,Y,Z)$). Podstawową rzeczą, jaką musimy zapewnić, jest poprawne przenoszenie „nadwyżkowych" cyfr. Reguła jest prosta – przechodzimy linię reprezentującą cyfry od najmniej znaczącego bitu w kierunku bardziej znaczących; gdy zaś natkniemy się na pozycję, na której obie liczby X i Y mają cyfrę 1, szukamy pierwszego miejsca, na którym obie te liczby mają cyfrę 0, po drodze wszędzie dodając cyfry „z nadwyżką":

$$\begin{aligned}\forall_x\Big(x \in X \land x \in Y \to \exists_{y>x}\Big(&y \notin X \land y \notin Y \land y \in Z \land \\ &\land \forall_z\big(x < z < y \to ((z \in X \land z \notin Y \land z \notin Z)\lor \\ &\qquad\lor(z \notin X \land z \in Y \land z \notin Z)\lor \\ &\qquad\lor(z \in X \land z \in Y \land z \in Z))\big)\Big)\Big)\end{aligned}$$

W pierwszej linii ustalamy pozycję x, na której X i Y mają jedynki i szukamy dla niej pierwszej pozycji y, na której obie te liczby mają zera. W obszarze pomiędzy zachodzi binarne dodawanie cyfr z nadwyżką. Zauważmy, że nie mówimy, jaką wartość ma przyjmować Z na pozycji x. Wynika to stąd, że nie wiemy, czy do tej pozycji nie „nadeszła" do nas nadwyżka przejęta z dodawania poprzednich cyfr.

Trzeba teraz poradzić sobie z fragmentami, do których nie dotarł „nadwyżkowy" bit. Jest to albo początkowy fragment zapisu (innymi słowy, na tej i żadnej wcześniejszej pozycji nie było jedynki jednocześnie w X i Y) lub fragment od pozycji, na której obie X i Y mają zera do pierwszego miejsca, gdzie obie mają jedynki. Oto owa formuła:

$$\begin{aligned}\forall_x\Big(\exists_{y\le x}\ \ (y = 0 \lor (y \notin X \land y \notin Y))\land \\ \land\forall_z(y \le z \le x \to &((z \notin X \land z \notin Y \land z \notin Z)\lor \\ &\lor(z \in X \land z \notin Y \land z \in Z)\lor \\ &\lor(z \notin X \land z \in Y \land z \in Z)))\Big)\end{aligned}$$

y równe 0 oznacza początek pierwszy z wymienionych przypadków, $y \notin X \land y \notin Y$ – drugi. Na całym obszarze od y do x tym razem dodajemy cyfry normalnie.

Konstrukcja formuły dla drzewa jest analogiczna; trzeba tylko pamiętać o tym, że „początkiem" będzie już nie element, który nie ma poprzednika, lecz taki, który jest lewym synem swego ojca (pozostałe wierzchołki będą synami prawymi). ☐

Definicja 26. *Niech M – automat ze stosem rozpoznający słowa nad alfabetem jednoliterowym. Rozważmy graf konfiguracji:*

$$G_M = \langle V = Q \times \Gamma^*, E\rangle$$

gdzie pierwsza współrzędna (z Q) oznacza stan, druga (z Γ^*) symbolizuje zawartość stosu, zaś E jest zdefiniowane następująco:

$$E = \{((q,v),(p,w)) : \textit{maszyna przechodzi z } (q,v) \textit{ do } (p,w) \textit{ czytając jedną literę}\}$$

Twierdzenie 46. *Dla każdego automatu ze stosem M graf G_M ma rozstrzygalną teorię MSO.*

Dowód. Ponieważ tym razem mamy do czynienia z teorią MSO, niezbędna będzie interpretacja element-element. Narzucającą się reprezentacją stosu jest włożenie grafu G_M w drzewo k-arne, gdzie k to wielkość alfabetu stosu (Γ). Musimy jednak jeszcze w jakiś sposób reprezentować aktualny stan. Jako że stanów mamy skończenie wiele, pomysł polega na zwiększeniu arności drzewa o 1, numerowanie alfabetu stosu

wyłącznie liczbami dodatnimi i reprezentowanie stanu q_i przez i kolejnych zer (oczywiście wymaga to najpierw ponumerowania stanów). Mamy więc włożenie elementów V:

$$(\langle q_i, a_{i_1} a_{i_2} \dots a_{i_k} \rangle) = i_1 i_2 \dots i_k 0^i$$

Włożenie zbioru krawędzi E jest wyznaczone jednoznacznie przez włożenie zbioru wierzchołków V. Należenie do obrazu $i[V]$ definiuje poniższa pseudodefinicja:

$$\varphi_V(x) \quad \equiv \quad x \in \bigcup_{1 \leq i \leq m} \{1, 2, \dots, n\}^* 0^i$$

gdzie n jest objętością alfabetu stosu, zaś m – ilością stanów. Oczywiście łatwo ją sformalizować w $MSO(\leq)$ (z predykatem $i(z)$ mówiącym, że na pozycji z występuje znak i):

$$\varphi_V(x) \quad \equiv \quad \exists_{y \in x} \quad \left(\left(\forall_{z \in x} \;\; z < y \rightarrow \bigvee_{i \in \{1,2,\dots,n\}} i(z) \right) \wedge (\forall_{z \in x} \;\; z \geq y \rightarrow 0(z)) \right)$$

Pozostała interpretacja krawędzi $\varphi_E(x, y)$. Zauważmy, że opisuje ona własność „lokalną" – stanów jest skończenie wiele, więc zmiany pomiędzy nimi można zapewnić formułami z wspólnie ograniczoną liczbą kwantyfikatorów, zaś w jednym kroku możemy przeczytać jedynie najwyższy symbol na stosie. Zamiast wyjściowego zbioru E można więc rozpatrywać:

$E' = \{((q, a), (p, k, w))$: maszyna po przeczytaniu jednej litery przechodzi ze stanu q, gdy znak na szczycie stosu to a do stanu p, zdejmując k znaków i dodając słowo $w\}$

Taki sposób zapisu bardziej akcentuje zmiany pomiędzy poszczególnymi stanami automatu, niż same stany; to z kolei łatwiej już zapisać w postaci formuły. Odpowiednia „pseudoformuła" (po ponumerowaniu stanów i alfabetu stosu) wygląda tak:

$\varphi_E(x, y) \equiv \bigvee_{((q_i, a_i), (q_j, k, w)) \in E'}$ „część dodatnia x kończy się na i $\wedge$

$\wedge$ „część zerowa x to 0^i, część zerowa y to 0^j" $\wedge$

$\wedge$ „część dodatnia y to skrócona o k końcowych znaków część dodatnia x powiększona o słowo w"

Dokładne przedstawienie powyższej formuły pozostawiam jako ćwiczenie czytelnikowi. □

Wnioski: Następujące trzy problemy są rozstrzygalne:

Twierdzenie 47. *Dane: automat ze stosem, konfiguracja (q, v).*
Teza: czy konfiguracja ta jest osiągalna?

Dowód. Rozwiązanie: zgadujemy zbiór konfiguracji i sprawdzamy, czy tworzy on ścieżkę z konfiguracji początkowej. □

Twierdzenie 48. *Dane: automat ze stosem.*
Teza: czy istnieje nieskończenie wiele konfiguracji osiągalnych?

Dowód. Postępujemy podobnie jak przy dowodzie twierdzenia 45 – zbiór konfiguracji osiągalnych jest nieskończony wtedy i tylko wtedy, gdy zbiór ich przodków (w sensie relacji E) w ścieżkach bezcyklowych jest nieskończony. Stąd i z lematu Königa wiemy, że w takiej sytuacji istnieje bieg nieskończony bez cyklu, wystarczy więc go zgadnąć i sprawdzić, że jest poprawny. □

Twierdzenie 49. *Dane: automat ze stosem.*
Teza: czy istnieje bieg nieskończony, który ma dowolnie głębokie stosy?

Dowód. Bieg nieskończony o dowolnie głębokich stosach istnieje wtedy i tylko wtedy, gdy istnieje nieskończony bieg bezcyklowy. Jest to zatem problem identyczny z poprzednim. □

Do tej pory rozważaliśmy głównie formuły w uniwersum konkretnego grafu nieskończonego. Można również rozważać klasy grafów skończonych, w których spełnialność dla MSO jest rozstrzygalna.

Definicja 27. *Ustalmy k. Konstruujemy grafy nieskierowane (dopuszczamy pętle jednoelementowe), których wierzchołki są pokolorowane na jeden z kolorów ze zbioru $\{1, 2, \dots, k\}$. Zaczynamy od grafu pustego; mamy do dyspozycji następujące operacje:*

- *0-argumentowe:*
  - $\bullet i$*: tworzy jednowierzchołkowy graf z wierzchołkiem o kolorze i*
- *1-argumentowe:*
  - $\mathrm{dodaj}(i, j)$*: dodaje krawędzie pomiędzy wszystkimi parami wierzchołków, z których jeden ma kolor i, zaś drugi kolor j*
  - $\mathrm{zamien}(i, j)$*: koloruje wszystkie wierzchołki o kolorze i na kolor j*
- *2-argumentowe:*
  - $G_1 \uplus G_2$*: tworzy sumę rozłączną grafów G_1 i G_2*

Minimalne k, dla jakiego da się w ten sposób skonstruować graf G nazywa się szerokością klikową grafu G.

Twierdzenie 50. *Ustalmy $k \in \mathbb{N}$. Rozstrzygalny jest problem:*
Dane: $\varphi \in MSO(E)$
Teza: czy istnieje skończony graf nieskierowany G o szerokości klikowej $\leq k$ taki, że $G \models \varphi$?

Przykład.

- Przy $k = 1$ możemy jedynie uzyskać skończone sumy klik z antykliką. Polecenie $\mathrm{zamien}(i, j)$ nie ma sensu przy $k = 1$, zaś każde użycie $\mathrm{dodaj}(1, 1)$ zmienia aktualny graf w klikę. Jeśli nie użyjemy $\mathrm{dodaj}(1, 1)$, pozostaje nam jedynie operacja $\bullet i$, za pomocą której stworzymy dowolnie dużą antyklikę. Suma rozłączna dwóch grafów powyższej postaci też jest oczywiście powyższej postaci. To ostatecznie dowodzi, że żadnych innych grafów nie otrzymamy.

- Jak stworzyć linię długości n? (przez linię rozumiem pewien początkowy podzbiór zbioru liczb naturalnych z krawędziami pomiędzy elementami różniącymi się o 1) Okazuje się, że wystarczą do tego trzy kolory, niezależnie od wielkości n. Początek (linię jednopunktową) uzyskujemy łatwo, kolorując wierzchołek na kolor 2 (operacja $\bullet 2$). Załóżmy, że mamy już linię długości m, której wszystkie wierzchołki poza największym są koloru 1, zaś ów największy – koloru 2 lub 3; dla ustalenia uwagi załóżmy, że jest to 2. Postępujemy następująco – dodajemy nowy wierzchołek koloru 3 ($\bullet 3$), łączymy ów nowy punkt z jego poprzednikiem ($\mathrm{dodaj}(2, 3)$), a następnie zamieniamy kolor poprzednika na 1 ($\mathrm{zamien}(2, 1)$).

- Zauważmy, że gdybyśmy wyeliminowali możliwość istnienia pętli jednoelementowych, to w poprzednim przykładzie wystarczyłyby dwa kolory (zamiast 2 i 3 używalibyśmy wszędzie 2).

Dowód twierdzenia 50. Interpretacją będzie dla nas funkcja $i \colon T \mapsto G$, gdzie T jest drzewem, zaś G – grafem. Zauważmy, że każda konstrukcja grafu za pomocą operacji zdefiniowanych w definicji szerokości klikowej da się przedstawić na drzewie skończonym, którego każdy wierzchołek ma tyle synów, ile wskazuje arność operacji, jaką on reprezentuje (w szczególności operacje $\bullet i$ to dokładnie liście). Przyjmijmy następującą definicję:

Definicja 28. *Drzewo opisujące k-derywację to skończone drzewo etykietowane operacjami z definicji szerokości klikowej (przy $i, j \in \{1, 2, \dots, k\}$.*

(Uwaga: dla ustalonego k zbiór k-derywacji jest regularnym językiem drzew L_k)
[cdn...] □

Rozdział 9

Rozwiązania zadań

Zadanie 1.7. Napisać formuły FOL (logiki pierwszego rzędu) definiujące następujące własności:

a istnieje dokładnie k elementów w uniwersum,

b pomiędzy wierzchołkami u, v istnieje ścieżka długości $\leq k$,

c w uniwersum nie ma cyklu długości k,

Czy można napisać także formuły definiujące następujące własności (na pierwszy rzut oka prostsze wersje podpunktów wyżej)?

a' uniwersum ma parzyście wiele elementów,

b' istnieje ścieżka między wierzchołkami u i v,

c' w uniwersum nie istnieje cykl.

Rozwiązanie. (Zapisał Mateusz Kopeć.)

Niech $E(u,v) \iff$ między u i v istnieje krawędź.

a

$$\exists_{x_1,\ldots,x_k}\{(\bigwedge_{\substack{1\leq i,j\leq k\\ i\neq j}} x_i \neq x_j) \wedge (\forall_y \bigvee_{1\leq i\leq k} y = x_i)\}$$

b

$$E(u,v) \vee \bigvee_{1\leq i<k} \{\exists_{x_1,x_2,\ldots,x_i} E(u,x_1) \wedge E(x_i,v) \wedge \bigwedge_{1\leq j<i} E(x_j,x_{j+1})\}$$

c

$$\neg\exists_{x_1,x_2,\ldots,x_k}\{E(x_k,x_1) \wedge \bigwedge_{1\leq i<k} E(x_i,x_{i+1})\}$$

W logice pierwszego rzędu nie da się napisać formuły definiującej pozostałe własności. Aby to wykazać, skorzystamy z twierdzenia o zwartości:

Twierdzenie 51. *Nieskończony zbiór rachunku predykatów jest spełnialny jeśli tylko jego każdy podzbiór skończony jest spełnialny. Równoważnie, jeśli taki zbiór jest sprzeczny, to istnieje jego skończony podzbiór, który jest sprzeczny.*

a' Załóżmy, że istnieje formuła φ definiująca własność **a'**. Wtedy niech:

$$\varphi_k = \neg\exists_{x_1,\ldots,x_k}\{(\bigwedge_{\substack{1\leq i,j\leq k\\ i\neq j}} x_i \neq x_j) \wedge (\forall_y \bigvee_{i} y = x_i)\}$$

φ_k - formuła mówiąca, że uniwersum nie ma k elementów
φ - formuła mówiąca, że uniwersum ma parzyście wiele elementów
Korzystając z twierdzenia o zwartości dla zbioru predykatów

$$\{\varphi, \varphi_2, \varphi_4, \varphi_6, \varphi_8, \ldots\}$$

wiemy, że cały nieskończony zbiór predykatów jest spełnialny. Jest to sprzeczność, ponieważ uniwersum ma parzyście wiele elementów i jednocześnie wykluczyliśmy wszystkie skończone parzyste liczebności uniwersum za pomocą formuł φ_{2k}, gdzie $k \in \mathbb{N}$. Zatem w FOL nie istnieje formuła definiująca własność **a'**.

b' Załóżmy, że istnieje formuła φ definiująca własność **b'**. Wtedy niech:

$$\varphi_k = \neg E(u,v) \wedge \neg \bigvee_{1 \leq i < k} \{\exists_{x_1,x_2,\ldots,x_i} E(u,x_1) \wedge E(x_i,v) \wedge \bigwedge_{1 \leq j < i} E(x_j,x_{j+1})\}$$

φ_k - formuła mówiąca, że nie istnieje ścieżka długości $\leq k$ z u do v
φ - formuła mówiąca, że istnieje ścieżka z u do v
Korzystając z twierdzenia o zwartości dla zbioru predykatów

$$\{\varphi, \varphi_1, \varphi_2, \varphi_3, \varphi_4, \ldots\}$$

wiemy, że cały nieskończony zbiór predykatów jest spełnialny. Jest to sprzeczność, ponieważ istnieje ścieżka z u do v i jednocześnie wykluczyliśmy wszystkie ścieżki o skończonych długościach za pomocą formuł φ_k, gdzie $k \in \mathbb{N}$. Zatem w FOL nie istnieje formuła definiująca własność **b'**.

c' Załóżmy, że istnieje formuła φ definiująca własność **c'**. Wtedy niech:

$$\varphi_k = \neg \exists_{x_1,x_2,\ldots,x_k} \{E(x_k,x_1) \wedge \bigwedge_{1 \leq i < k} E(x_i,x_{i+1})\}$$

φ_k - formuła mówiąca, że nie istnieje cykl długości k
φ - formuła mówiąca, że nie istnieje cykl
Korzystając z twierdzenia o zwartości dla zbioru predykatów

$$\{\neg\varphi, \varphi_1, \varphi_2, \varphi_3, \varphi_4, \ldots\}$$

wiemy, że cały nieskończony zbiór predykatów jest spełnialny. Jest to sprzeczność, ponieważ nie istnieje cykl (wynika to z φ) i jednocześnie wykluczyliśmy wszystkie cykle o skończonych długościach za pomocą formuł φ_k, gdzie $k \in \mathbb{N}$. Zatem w FOL nie istnieje formuła definiująca własność **c'**.

Zadanie 2.1. Pokazać, że każdy niedeterministyczny automat z warunkiem Mullera jest równoważny pewnemu NBA.

Rozwiązanie. (Zapisał Michał Pilipczuk.)

Niech $\mathcal{A}_M$ będzie automatem o stanach Q z warunkiem Mullera akceptującym język L. Warunek automatu $\mathcal{A}_M$ (oznaczony jako $\psi_{\mathcal{A}}$) jest boolowską kombinacją warunków typu Büchiego, czyli zdań φ_q mówiących „stan q wystąpił nieskończenie często". Rozważmy wszystkie wartościowania zdań φ_q i podzielmy je na dwie grupy: grupa tych, przy których $\psi_{\mathcal{A}}$ jest prawdziwe, i grupa tych, przy których jest fałszywe. Budowany automat NBA musi mieć tę własność, że akceptuje słowa w których wartościowanie $\{\varphi_q\}$ należy do tych, przy których $\psi_{\mathcal{A}}$ jest prawdziwe. Automat będzie więc niedeterministycznie zgadywał, które wartościowanie jest spełnione i je sprawdzał.

Niech więc $\tau : \{\varphi_q : q \in Q\} \to \{0,1\}$ będzie boolowskim wartościowaniem warunków φ_q, dla którego mamy zbudować automat NBA $\mathcal{A}_\tau$ akceptujący jedynie takie słowa nieskończone, w których wartościowanie $\{\varphi_q\}$ to po prostu τ. Niech $\mathcal{B}_{inf} = \tau^{-1}(1)$ oraz $\mathcal{B}_{fin} = \tau^{-1}(0)$. Poprzez utożsamienie stanu z warunkiem Büchiego dlań możemy rozumieć $\mathcal{B}_{inf}$ i $\mathcal{B}_{fin}$ jako odpowiednio zbiory stanów, w których automat musi się znaleźć nieskończenie wiele razy lub też skończenie wiele razy. Niech $\overline{\mathcal{A}_M}$ będzie automatem $\mathcal{A}_M$ z usuniętymi stanami z $\mathcal{B}_{fin}$. Automat $\mathcal{A}_\tau$ będzie składał się z jednej kopii automatu $\mathcal{A}_M$ oraz $|\mathcal{B}_{inf}|$ kopii automatu $\overline{\mathcal{A}_M}$, indeksowanych stanami z $\mathcal{B}_{inf}$: $\mathcal{A}_{q_1}$,$\mathcal{A}_{q_2}, \ldots$,$\mathcal{A}_{q_{|\mathcal{B}_{inf}|}}$. Automat $\mathcal{A}_\tau$ zaczyna w kopii $\mathcal{A}_M$ dokładnie tak jak $\mathcal{A}_M$ i kręci się tam ile chce. Ponadto z każdego stanu spoza $\mathcal{B}_{fin}$ za pomocą ε-przejścia może przeskoczyć do tego samego stanu w automacie $\mathcal{A}_{q_1}$. Następnie kręci się w $\mathcal{A}_{q_1}$, ale ze stanu q_2 za pomocą ε-przejścia może przeskoczyć do q_2 w automacie $\mathcal{A}_{q_2}$. Konstruujemy tak dalej, aż wreszcie ze stanu q_1 w automacie $\mathcal{A}_{q_{|\mathcal{B}_{inf}|}}$ za pomocą ε-przejścia przechodzi do q_1 w automacie $\mathcal{A}_{q_1}$. Stanem akceptującym w tym NBA jest sztuczny stan dodany wewnątrz któregokolwiek ε-przejścia pomiędzy q_i a q_{i+1}.

Łatwo sprawdzić, że automat tak skonstruowany spełnia nałożone nań warunki. Moment opuszczenia kopii $\mathcal{A}_M$ i przejścia do $\mathcal{A}_{q_1}$ odpowiada decyzji „od teraz nie będzie już stanów z $\mathcal{B}_{fin}$", co załatwiamy za pomocą usunięcia ich z automatów $\mathcal{A}_{q_i}$. Krążenie po pętli ($\mathcal{A}_{q_1}$,$\mathcal{A}_{q_2}, \ldots$,$\mathcal{A}_{q_{|\mathcal{B}_{inf}|}}$) odpowiada poszukiwaniu kolejno: następnego wystąpienia q_2, następnego wystąpienia q_3 itd., co odpowiada **dokładnie** warunkowi, że każdy z stan z $\mathcal{B}_{inf}$ wystąpił nieskończenie wiele razy. Umiejscowienie stanu akceptującego zapewnia nam nieskończenie wiele obrotów pętli.

Cały automat budujemy poprzez zbudowanie $\mathcal{A}_\tau$ dla każdego τ spełniającego warunki z pierwszego akapitu oraz podłączenie ε-przejściami ich stanów startowych do nowego, sztucznego stanu startowego. Jeśli nie lubimy ε-przejść, to dzięki niedeterminizmowi możemy je usunąć — w szczególności można to zrobić tak, by nie móc „omijać" stanów akceptujących automatów $\mathcal{A}_\tau$.

Zadanie 2.2. Pokazać, że języki rozpoznawane przez deterministyczne automaty z warunkiem Büchiego są zamknięte na przecięcie.

Rozwiązanie. (Zapisał Michał Pilipczuk.)

Niech $\mathcal{A}_1$ oraz $\mathcal{A}_2$ będą dwoma automatami DBA. Chcemy skonstruować jeden automat DBA rozpoznający $L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$. Automat ten (nazwijmy go A) będzie chodził po trzech kopiach produktu automatów $\mathcal{A}_1 \times \mathcal{A}_2$. Pierwsza kopia (oznaczana $\mathcal{B}_1$) będzie odpowiadała rozpoznawaniu wystąpień stanów akceptujących w $\mathcal{A}_1$, druga ($\mathcal{B}_2$) w $\mathcal{A}_2$. Trzecia kopia ($\mathcal{B}_F$) będzie kopią stanów akceptujących.

Przejścia w automacie $\mathcal{B}_1$ są takie same jak w standardowym automacie produktowym, oprócz przejść do stanów postaci (q,p) dla q akceptującego w $\mathcal{A}_1$. Zamiast odpowiedniego przejścia do (q,p) w $\mathcal{B}_1$ przejście to prowadzi do (q,p) w automacie $\mathcal{B}_F$. Ze stanu (q,p) w $\mathcal{B}_F$ idzie ε-przejście do (q,p) w $\mathcal{B}_2$.

Podobnie, w automacie $\mathcal{B}_2$ przejścia do stanów postaci (q,p) dla p akceptującego w $\mathcal{A}_2$ zamieniamy na przejścia do stanu (q,p) w kopii $\mathcal{B}_1$.

Zauważmy, że usuwając ε-przejścia w tak skonstruowanym automacie uzyskamy automat DBA, gdyż jedyne ε-przejścia prowadziły ze stanów (q,p) w $\mathcal{B}_F$ do (q,p) w $\mathcal{B}_2$ i dla każdego (q,p) jest tylko jedno takie przejście. Skoro stany w $\mathcal{B}_2$ na mocy konstrukcji mają deterministyczne wyjścia, to po usunięciu ε-przejść również stany z $\mathcal{B}_F$ będą miały deterministyczne wyjścia.

Automat ten oczywiście rozpoznaje język $L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$. Bieg akceptujący w nim nieskończenie często przechodzi przez akceptującą część $\mathcal{B}_F$, co oznacza, że wykonuje nieskończenie wiele pętli pomiędzy $\mathcal{B}_1$ a $\mathcal{B}_2$. Jednocześnie każdy bieg robiący nieskończenie wiele takich pętli przechodzi przez $\mathcal{B}_F$. Każdorazowe przejście z $\mathcal{B}_1$ do $\mathcal{B}_2$ lub z $\mathcal{B}_2$ do $\mathcal{B}_1$ odpowiada wystąpieniu stanu akceptującego w $\mathcal{A}_1$ lub $\mathcal{A}_2$ odpowiednio. Zatem biegi akceptujące w tym automacie odpowiadają dokładnie biegom akceptowanym zarówno przez $\mathcal{A}_1$ jak i $\mathcal{A}_2$.

Zadanie 2.3. Pokazać, że następujący język nie jest ω-regularny:

$$\{a^{n_1}ba^{n_2}b\cdots : \lim_{i\to\infty} n_i = \infty\}.$$

Rozwiązanie. (Zapisał Michał Pilipczuk.)

Załóżmy przeciwnie, że język ten jest ω-regularny, jest zatem rozpoznawany przez pewien automat NBA o n stanach. Rozważmy bieg tego automatu rozpoznający słowo $abaabaaab\ldots$ (w każdym kolejnym bloku jest o jedno więcej a). Rozważmy blok a^k dla $k > n$. Z zasady szufladkowej Dirichleta któryś stan się powtórzył przy rozpoznawaniu tego bloku, mamy więc cykl w biegu. Załóżmy, że na cyklu tym pojawił się stan akceptujący. Wówczas pompując cykl w nieskończoność uzyskujemy, że również słowo $abaabaaab\ldots a^{k-1}ba^{\omega}$ należy do języka, co jest nieprawdą. Stąd wszystkie cykle wewnątrz bloków postaci a^k przechodzą jedynie przez stany nieakceptujące. Usuwając je nie zmieniamy zatem warunku Büchiego, czyli po usunięciu słowo nadal będzie w języku. Ze względu na rozmiar automatu możemy w każdym bloku wybierać cykl i usuwać aż do momentu, kiedy jego długość będzie nie większa niż n. Uzyskujemy wniosek, że do języka należy słowo postaci $a^{k_1}ba^{k_2}ba^{k_3}b\ldots$ dla $k_i \le n$, co stoi w sprzeczności z warunkiem $k_i \to \infty$. W obu przypadkach uzyskaliśmy więc sprzeczność, stąd język ten nie jest ω-regularny.

Zadanie 2.4. Słowo ostatecznie okresowe to słowo postaci wv^ω dla $w,v \in A^+$. Pokazać, że jeśli dwa języki ω-regularne zawierają te same słowa ostatecznie okresowe, to muszą być równe.

Rozwiązanie. (Zapisał Krzysztof Kąs.)

Niech $\mathcal{A}, \mathcal{B}$ będą automatami niedeterministycznymi z warunkiem Büchiego rozpoznającymi te języki.

Przypuśćmy, że rozważane języki nie są równe. Bez straty ogólności załóżmy więc, że istnieje słowo nieskończone $w = a_0a_1a_2a_3\ldots$, które należy do języka rozpoznawanego przez automat $\mathcal{B}$, ale nie należy do języka rozpoznawanego przez automat $\mathcal{A}$.

Dla dowolnego słowa skończonego $x \in A^+$ przez $M_x^{\mathcal{A}}$ i $M_x^{\mathcal{B}}$ oznaczmy macierze odpowiadające słowu x w automatach $\mathcal{A}$ i $\mathcal{B}$ odpowiednio.

Rozważmy podział $w = v_0v_1v_2v_3\ldots$ i macierze $M_{\mathcal{B}}, N_{\mathcal{B}}$ otrzymane z zastosowania kluczowego lematu. Wówczas:

- $M_{v_0}^{\mathcal{B}} = M_{\mathcal{B}}$
- $M_{v_i}^{\mathcal{B}} = N_{\mathcal{B}}$ dla $i = 1, 2, 3, \ldots$
- $M_{\mathcal{B}} \cdot N_{\mathcal{B}} = M_{\mathcal{B}}$ i $N_{\mathcal{B}} \cdot N_{\mathcal{B}} = N_{\mathcal{B}}$.

Rozważmy teraz ten sam podział w kontekście automatu $\mathcal{A}$. Otrzymamy ciąg macierzy:

$$M_{v_i}^{\mathcal{A}} \text{ dla } i = 0, 1, 2, 3, \ldots.$$

Ponownie stosujemy kluczowy lemat, ale tym razem dla trochę innych danych:

- Zamiast automatu $\mathcal{B}$ rozważamy automat $\mathcal{A}$.
- Zamiast liter $a_0, a_1, a_2, a_3, \ldots$ rozważamy słowa $v_0, v_1, v_2, v_3, \ldots$. Rozważanie słów zamiast pojedynczych liter nie ma wpływu na poprawność dowodu kluczowego lematu. Wystarczy bowiem w definicji α zamiast macierzy $M_{a_i \ldots a_{j-1}}$ rozważać macierze $M_{v_i \ldots v_{j-1}}$.

Otrzymamy w ten sposób podział $w = u_0u_1u_2u_3\ldots$, gdzie $u_i = v_{s_i}v_{s_i+1}\ldots v_{s_{i+1}-1}$ dla pewnego ciągu liczb naturalnych $0 = s_0 < s_1 < s_2 < \ldots$. Kluczowy lemat wyznaczy też macierze $M_{\mathcal{A}}$ i $N_{\mathcal{A}}$ takie, że:

- $M_{u_0}^{\mathcal{A}} = M_{\mathcal{A}}$
- $M_{u_i}^{\mathcal{A}} = N_{\mathcal{A}}$ dla $i = 1, 2, 3, \ldots$
- $M_{\mathcal{A}} \cdot N_{\mathcal{A}} = M_{\mathcal{A}}$ i $N_{\mathcal{A}} \cdot N_{\mathcal{A}} = N_{\mathcal{A}}$.

Jednocześnie nadal w automacie $\mathcal{B}$ mamy:

- $M_{u_0}^{\mathcal{B}} = M_{v_0}^{\mathcal{B}} \cdot M_{v_1}^{\mathcal{B}} \cdot M_{v_2}^{\mathcal{B}} \cdot \ldots \cdot M_{v_{s_1-1}}^{\mathcal{B}} = M_{\mathcal{B}} \cdot N_{\mathcal{B}} \cdot N_{\mathcal{B}} \cdot \ldots \cdot N_{\mathcal{B}} = M_{\mathcal{B}}$
- $M_{u_i}^{\mathcal{B}} = M_{v_{s_i}}^{\mathcal{B}} \cdot M_{v_{s_i+1}}^{\mathcal{B}} \cdot M_{v_{s_i+2}}^{\mathcal{B}} \cdot \ldots \cdot M_{v_{s_{i+1}-1}}^{\mathcal{B}} = N_{\mathcal{B}} \cdot N_{\mathcal{B}} \cdot N_{\mathcal{B}} \cdot \ldots \cdot N_{\mathcal{B}} = N_{\mathcal{B}}$ dla $i = 1, 2, 3, \ldots$
- $M_{\mathcal{B}} \cdot N_{\mathcal{B}} = M_{\mathcal{B}}$ i $N_{\mathcal{B}} \cdot N_{\mathcal{B}} = N_{\mathcal{B}}$.

Rozważmy słowo $\tilde{w} = u_0u_1^\omega = u_0u_1u_1u_1\ldots$. Wielokrotnie korzystając z lematu charakteryzującego otrzymujemy kolejno:

- $\mathcal{B}$ akceptuje w,
- macierze $M_{\mathcal{B}}, N_{\mathcal{B}}$ (odpowiadające podziałowi $w = u_0u_1u_2u_3\ldots$) spełniają warunek $(\star)$,
- macierze $M_{\mathcal{B}}, N_{\mathcal{B}}$ (odpowiadające podziałowi $\tilde{w} = u_0u_1u_1u_1\ldots$) spełniają warunek $(\star)$,
- $\mathcal{B}$ akceptuje $\tilde{w} = u_0u_1^\omega$,
- (z założenia zadania) $\mathcal{A}$ akceptuje $\tilde{w} = u_0u_1^\omega$,

- macierze $M_{\mathcal{A}}, N_{\mathcal{A}}$ (odpowiadające podziałowi $\tilde{w} = u_0 u_1 u_1 u_1 \ldots$) spełniają warunek ($\star$),
- macierze $M_{\mathcal{A}}, N_{\mathcal{A}}$ (odpowiadające podziałowi $w = u_0 u_1 u_2 u_3 \ldots$) spełniają warunek ($\star$),
- $\mathcal{A}$ akceptuje w, co daje sprzeczność.

Prostsze rozwiązanie. (Zapisał Paweł Parys.)

Oznaczmy te języki przez L_1, L_2. Niech $L = L_1 - L_2$; jest to język ω-regularny, gdyż klasa języków ω-regularnych jest zamknięta na operacje boolowskie. Rozważmy NBA $\mathcal{A}$ rozpoznający L. Jeśli L jest niepusty, to w $\mathcal{A}$ istnieje ścieżka do pewnego stanu akceptującego oraz pewien cykl zawierający ten stan. Idąc tą ścieżką, a następnie w nieskończoność tym cyklem, dostajemy bieg akceptujący pewne słowo ostatecznie okresowe. Jednak z założenia L nie zawiera słów ostatecznie okresowych, czyli L jest pusty. Przez symetrię także $L_2 - L_1$ jest pusty, czyli $L_1 = L_2$.

Zadanie 2.5. Wskazać język ω-regularny, który nie jest różnicą dwóch języków rozpoznawanych przez deterministyczne automaty z warunkiem Büchiego.

Rozwiązanie. (Zapisał Mateusz Kopeć.)

Zauważmy, że wystarczy znaleźć język regularny L, który nie jest rozpoznawany przez zdefiniowaną poniżej klasę automatów. (ta klasa automatów rozpoznaje języki postaci $L_1 \setminus L_2$, gdzie L_1, L_2 są rozpoznawane przez deterministyczne automaty Büchiego)

$$\mathcal{A} =< \Sigma, Q, q_0, \delta, F, Z >$$

Bieg $q_0, q_1, \ldots$ jest akceptujący wtedy i tylko wtedy, gdy

$$Inf(q_0, q_1, \ldots) \cap F \neq \emptyset \wedge Inf(q_0, q_1, \ldots) \cap Z = \emptyset$$

gdzie Inf - zbiór stanów, pojawiających się nieskończenie wiele razy w biegu $q_0, q_1, \ldots$.
Załóżmy, że ten automat potrafi rozpoznawać język regularny $L =$ 'nieskończenie wiele a lub skończenie wiele b' (nad alfabetem $\{a,b,c\}$). 'Oszukamy' ten automat, podobnie jak postępowaliśmy podczas 'oszukiwania' deterministycznego automatu Büchiego:

- Rozpoczynamy od ciągu liter c, aż do momentu, gdy automat wejdzie do stanu $q \in F$,
- wstawiamy literę a,
- wpisujemy ciąg liter b, do momentu, gdy automat wejdzie do stanu $p \in Z$,
- powtarzamy postępowanie od początku.

W ten sposób napisaliśmy słowo $(c^*ab^*)^\omega$, które należy do języka $L =$ 'nieskończenie wiele a lub skończenie wiele b', a nie zostało zaakceptowane przez nasz automat (ponieważ nieskończenie często weszliśmy do stanów ze zbioru Z). Ze sprzeczności wynika, że żaden automat z wyżej zdefiniowanej klasy nie jest w stanie rozpoznać języka L, zatem nie jest to język postaci $L_1 \setminus L_2$, gdzie L_1, L_2 są rozpoznawane przez deterministyczne automaty Büchiego.

Zadanie 2.6. Zaprojektować algorytm, który sprawdza czy język rozpoznawany przez dany NBA jest przeliczalny.

Rozwiązanie. (Zapisał Krzysztof Kąs.)

Na początku tworzymy deterministyczny automat $\mathcal{A}$ z warunkiem Mullera równoważny danemu NBA. Niech zbiorem jego stanów będzie Q, natomiast stanem początkowym - q_I. Niech podzbiory $F_1, F_2, ..., F_j \subseteq Q$ określają warunek akceptacji:

> $\mathcal{A}$ akceptuje słowo nieskończone w wtw. zbiór stanów występujących nieskończenie często w biegu automatu $\mathcal{A}$ na słowie w jest jednym ze zbiorów $F_1, F_2, ..., F_j$.

Język rozpoznawany przez automat jest przeliczalny wtedy i tylko wtedy gdy dla każdego $i \in \{1,2,\ldots,j\}$ przeliczalny jest zbiór słów w, dla których:

> zbiór stanów występujących nieskończenie często w biegu automatu $\mathcal{A}$ na słowie w jest równy F_i.

Zatem wystarczy podać algorytm w przypadku gdy $j = 1$. Dla uproszczenia dalszego zapisu oznaczmy $F = F_1$. Niech $F = \{f_1, f_2, \ldots, f_k\}$.

Mając dany taki automat $\mathcal{A}$ tworzymy graf skierowany G w następujący sposób:

- wierzchołkami są stany automatu $\mathcal{A}$,
- oznaczenia q_I, Q, F przenoszą się bezpośrednio na wierzchołki grafu G,
- z wierzchołka q_a do wierzchołka q_b istnieje krawędź wtedy i tylko wtedy, gdy w automacie $\mathcal{A}$ można było przejść po jednej literze ze stanu q_a do stanu q_b.

Okazuje się, że dany problem można teraz rozstrzygnąć patrząc tylko na graf G.

Niech H będzie podgrafem grafu G indukowanym przez zbiór F. Język rozpoznawany przez automat $\mathcal{A}$ jest przeliczalny wtw. gdy spełniony jest co najmniej jeden spośród warunków:

- w grafie G nie istnieje ścieżka z q_I do żadnego wierzchołka ze zbioru F,
- graf H nie jest ściśle spójny (tzn. istnieją wierzchołki $f_\alpha, f_\beta \in F$ takie, że nie istnieje w H ścieżka z f_α do f_β),
- graf H jest pojedynczym cyklem.

Pierwszy z warunków oznacza bowiem, że język rozpoznawany przez automat $\mathcal{A}$ jest pusty (a więc przeliczalny).

Drugi warunek też implikuje pustość tego języka. Dla dowodu zauważmy, że wówczas istnieją takie wierzchołki $f_\alpha, f_\beta \in F$, że każda ścieżka z f_α do f_β w G przechodzi przez jakiś wierzchołek spoza zbioru F. Gdyby więc automat akceptował pewne słowo w, to jego bieg na tym słowie nieskończenie często przechodziłby przez f_α i nieskończenie często przez f_β, a więc nieskończenie często musiałby przechodzić ścieżką z f_α do f_β przez wierzchołki spoza zbioru F. Daje to sprzeczność, bo każdy bieg akceptujący automatu $\mathcal{A}$ od pewnego momentu odwiedza tylko stany ze zbioru F.

Trzeci warunek oznacza, że każdy bieg akceptujący $\mathcal{A}$ na pewnym słowie w od pewnego momentu porusza się zgodnie z cyklem H. Wobec tego $w = uv^\omega$ dla pewnych słów skończonych u, v. A jak wiadomo słów tej postaci jest przeliczalnie wiele.

Pozostaje do pokazania, że niespełnienie tych trzech warunków implikuje nieprzeliczalność danego języka.

Wówczas graf H ma więcej niż F krawędzi. Zatem istnieje w grafie H wierzchołek f_γ oraz ścieżka z f_γ do f_γ długości mniejszej niż $|F|$ i większej od zera. Z silnej spójności wynika, że istnieje też ścieżka z f_γ do f_γ przechodząca przez wszystkie wierzchołki (jej długość jest oczywiście nie mniejsza niż $|F|$). Zatem istnieją:

- słowo skończone u,

- słowo skończone x długości nie mniejszej niż $|F|$,
- niepuste słowo skończone y długości mniejszej niż $|F|$,

takie, że:

- automat $\mathcal{A}$ ze stanu q_I przechodzi do stanu f_γ po słowie u,
- automat $\mathcal{A}$ ze stanu f_γ przechodzi do stanu f_γ po słowie x, odwiedzając po drodze wszystkie stany ze zbioru F,
- automat $\mathcal{A}$ ze stanu f_γ przechodzi do stanu f_γ po słowie y (ale nie odwiedza po drodze wszystkich stanów ze zbioru F).

Rozważmy słowa $z = x^{|y|}$ oraz $t = y^{|x|}$. Wówczas $|z| = |t|$. Ponadto nadal:

- automat $\mathcal{A}$ ze stanu q_I przechodzi do stanu f_γ po słowie u,
- automat $\mathcal{A}$ ze stanu f_γ przechodzi do stanu f_γ po słowie z, odwiedzając po drodze wszystkie stany ze zbioru F,
- automat $\mathcal{A}$ ze stanu f_γ przechodzi do stanu f_γ po słowie t (ale nie odwiedza po drodze wszystkich stanów ze zbioru F).

Ponieważ automat $\mathcal{A}$ jest deterministyczny, to $z \neq t$. Wobec tego język rozpoznawany przez automat $\mathcal{A}$ zawiera następujący język nieprzeliczalny:

$$u(z(z+t))^\omega.$$

Zadanie 4.1. Przez WMSO, z angielskiego Weak Monadic Second-Order Logic, oznaczamy wariant MSO, w którym wolno kwantyfikować wyłącznie po elementach oraz skończonych podzbiorach uniwersum.

Pokazać, że każda formuła WMSO($\leq$) na słowach nieskończonych jest równoważna formule logiki MSO($\leq$).

Rozwiązanie. (Zapisał Michał Gołębiowski.)

($\subseteq$) Pokażemy najpierw, że każda formuła $WMSO(s)$ da się zapisać jako formuła $MSO(s)$. Rozpatrzmy formułę w logice, w której dopuszczamy kwantyfikatory obu logik $WMSO(s)$ i $WMSO(s)$:

$$\forall_{X-\text{sk.}}\ \varphi(X) \tag{9.1}$$

gdzie $\varphi(X)$ jest formułą $MSO(s)$. Jeśli w $\varphi(X)$ występują zmienne wolne, traktujemy je jako stałe. Zdefiniujmy formułę $MSO(s)$:

$$\forall_y \forall_X\ ((\forall_x\ (X(x)\ \rightarrow\ x \leq y))\ \rightarrow\ \varphi(X)) \tag{9.2}$$

Musimy udowodnić, że formuła (9.2) jest prawdziwa wtedy i tylko wtedy, gdy dla każdego skończonego X zachodzi $\varphi(X)$. Załóżmy, że X jest skończony; udowodnimy, że formuła (9.2) implikuje $\varphi(X)$. Każdy skończony zbiór pozycji w biegu automatu jest ograniczony z góry, istnieje więc y będące ograniczeniem górnym dla wszystkich elementów zbioru X. Ustalmy takie y. Ponieważ formuła:

$$(\forall_x\ (X(x)\ \rightarrow\ x \leq y))\ \rightarrow\ \varphi(X) \tag{9.3}$$

ma być prawdziwa dla dowolnego y, to i również dla wybranego przez nas. Z wyboru y wynika bezpośrednio, że lewa strona implikacji jest prawdziwa. Ponieważ z założenia cała implikacja jest prawdziwa, to zachodzi $\varphi(X)$, co chcieliśmy udowodnić.

Załóżmy teraz, że dla każdego skończonego X zachodzi $\varphi(X)$. Wówczas dla każdego X i y albo y nie jest ograniczeniem górnym dla X i wtedy przesłanka implikacji (9.3) jest prawdziwa, albo y jest ograniczeniem górnym X, a zatem X jest skończony; zachodzi zatem teza implikacji (9.3).

Podobnie dla formuły:

$$\exists_{X-\text{sk.}}\ \varphi(X) \tag{9.4}$$

gdzie $\varphi(X)$ jest formułą $MSO(s)$, definiujemy zdanie w całości w $MSO(s)$:

$$\exists_y \exists_X\ ((\forall_x\ (X(x)\ \rightarrow\ x \leq y))\ \wedge\ \varphi(X)) \tag{9.5}$$

Musimy pokazać, że formuła (9.5) jest prawdziwa wtedy i tylko wtedy, gdy dla pewnego skończonego X zachodzi $\varphi(X)$. Załóżmy najpierw, że (9.5) jest prawdziwa. Wybierzmy X i y realizujące prawdziwość formuły:

$$(\forall_x\ (X(x)\ \rightarrow\ x \leq y))\ \wedge\ \varphi(X)$$

Warunek $\forall_x\ (X(x)\ \rightarrow\ x \leq y)$ oznacza w szczególności, że X jest skończony; z drugiego z warunków wiemy, że dla tego skończonego X zachodzi $\varphi(X)$.

Niech teraz istnieje skończony zbiór X, dla którego zachodzi $\varphi(X)$. Istnieje wówczas (na mocy poprzednich rozważań) jego ograniczenie górne y – ustalmy takie y. Wówczas zachodzi zarówno formuła:

$$\forall_x\ (X(x)\ \rightarrow\ x \leq y)$$

jak i $\varphi(X)$, co dowodzi prawdziwości formuły (9.5), QED.

($\supseteq$) Załóżmy teraz, że język słów nieskończonych L jest definiowany w $MSO(s)$. Na mocy tw. Büchiego wiemy, że L jest ω-regularny. Niech $\mathcal{A}$ będzie automatem deterministycznym z warunkiem parzystości (o stanach z Q i rangach z $\{0, 1, \ldots, n\}$ oraz wartościowaniu $\alpha\colon Q \rightarrow \{0, 1, \ldots, n\}$) rozpoznającym ten język. Dzięki temu, że automat jest deterministyczny, aby istniał nieskończony bieg, w którym największa występującą dowolnie daleko ranga jest parzysta potrzeba i wystarczy, by – po „wyrzuceniu" odpowiednio długiego prefiksu biegu automatu na słowie – istniały dowolnie długie skończone biegi tego automatu na prefiksie badanego słowa o tej własności, że kończą się one na stanie o randze parzystej, która jest jednocześnie maksymalną rangą występującą na owym skończonym biegu. Taką własność już w oczywisty sposób zapisuje

się w $WMSO(\leq)$. Podobnie jak w przypadku formuł $MSO(\leq)$ dla języków regularnych słów skończonych, oznaczamy przez X_i zbiór pozycji, na których jesteśmy w stanie $q_i \in Q = \{q_0, q_1, \ldots, q_{|Q|}\}$; formuła będzie postaci:

$$\bigvee_{\substack{k \in \{0,1,\ldots,n\} \\ 2|k}} \exists_x \forall_{y \geq x} \exists_{z \geq x} \exists_{X_0 - \text{sk.}} \exists_{X_1 - \text{sk.}} \ldots \exists_{X_{|Q|} - \text{sk.}} \ (\psi \wedge \phi)$$

gdzie ψ jest zdefiniowana jako koniunkcja następujących podformuł:

$$\psi = \begin{cases} \alpha_k(z) \\ \qquad \text{na pozycji } x \text{ ranga wynosi } k \\ \\ \forall_t \ \ (x \leq t \leq z \bigvee_{l \leq k} \ \ \alpha_l(t)) \\ \qquad \text{rangi na pozycjach od } x \text{ do } z \text{ nie przekraczają } k \end{cases}$$

zaś ϕ jest zdefiniowana tak, jak pierwsze cztery składniki formuły z dowodu twierdzenia Büchiego. Stąd logiki $WMSO(s)$ oraz $MSO(s)$ mają identyczną siłę wyrazu w uniwersum słów nieskończonych, QED.

Zadanie 4.4. Porównać powyższą topologię z topologią zadaną przez odległość:

$$e(w,v) = \frac{1}{n} \qquad \text{gdzie } n \text{ to najmniejsza ilość stanów w NBA, który akceptuje } w \text{ ale nie } v.$$

Rozwiązanie. (Zapisał Krzysztof Kąs.)

Niech $a, b \in A$ będą pewnymi różnymi symbolami alfabetu.

1. Dla dowolnych słów $w,v \in A^\omega$ zachodzi

$$\frac{1}{\frac{1}{d(w,v)} + 2} \leq e(w,v). \tag{9.6}$$

Dla dowodu zauważmy, że ten warunek jest równoważny następującemu:

> Jeśli n jest pierwszą pozycją, na której w, v się różnią (bez straty ogólności n-ta litera słowa w to a i n-ta litera słowa v to b), to istnieje NBA o $n+2$ stanach, który akceptuje w i nie akceptuje v.

Poprzednik tej implikacji mówi, że istnieje słowo skończone $u = c_1c_2c_3 \ldots c_{n-1}$ długości $n-1$ takie, że $w \in uaA^\omega$ i $v \in ubA^\omega$. Skonstruujemy NBA o $n+2$ stanach rozpoznający język uaA^ω w następujący sposób:

- $q_0, q_1, q_2, \ldots, q_{n+1}$ są stanami,
- q_0 jest stanem początkowym,
- q_n jest stanem akceptującym,
- relacja przejścia jest zdefiniowane następująco ($i = 0, 1, 2, \ldots, n-2$):

ze stanu q_i po literze c_{i+1} jest przejście do stanu q_{i+1},
ze stanu q_i po literze innej niż c_{i+1} jest przejście do stanu q_{n+1},
$q_{n-1} \xrightarrow{a} q_n$,
ze stanu q_{n-1} po literze innej niż a jest przejście do stanu q_{n+1},
$q_n \xrightarrow{A} q_n, \quad q_{n+1} \xrightarrow{A} q_{n+1}$ (czyli te stany są „czarnymi dziurami”).

Warunek (9.6) oznacza, że $\frac{d(w,v)}{2d(w,v)+1} \leq e(w,v)$, czyli w szczególności:

$$d(w,v) \leq 3e(w,v). \tag{9.7}$$

2. Metryki d i e dają różne topologie. Rozważmy słowo $u = a^\omega$ oraz ciąg słów $v_n = a^nb^\omega$. Wówczas oczywiście $d(u,v_n) = \frac{1}{n+1} \to 0$. Natomiast $e(u,v_n) = \frac{1}{2}$ (nie zbiega do 0). Wynika to z faktu, że minimalny automat rozróżniający u i v_n ma dwa stany: p (początkowy) i q (akceptujący). Jego przejściami są:

$$p \xrightarrow{a} p \qquad p \xrightarrow{b} q \qquad q \xrightarrow{a} q \qquad q \xrightarrow{b} q.$$

3. Przestrzeń (A^ω, d) jest zupełna, natomiast przesrzeń (A^ω, e) nie jest zupełna.

Najpierw pokażemy, że przestrzeń (A^ω, d) jest zupełna.

Jeśli mamy ciąg słów $w_n \in A^\omega$ spełniający warunek Cauchy'ego w metryce d, to istnieje ciąg rosnący $m_1 < m_2 < m_3 < \ldots$, taki, że dla każdego n i dla każdego $k > m_n$ zachodzi

$$d(w_{m_n}, w_k) < \frac{1}{n}.$$

Stąd wynika, że słowa $w_{m_n}, w_{m_n+1}, w_{m_n+2}, \ldots$ na pozycji n-tej mają taki sam symbol (oznaczmy go przez a_n). Łatwo zauważamy, że słowo $a = a_1a_2a_3a_4\ldots$ dla każdego n ma na pozycji n-tej ten sam symbol co słowa $w_{m_n}, w_{m_n+1}, w_{m_n+2}, \ldots$. Stąd wynika, że dla każdego n i dla każdego $k \geq m_n$ zachodzi

$$d(a, w_k) < \frac{1}{n}.$$

Wobec tego a jest granicą ciągu w_n w metryce d. Czyli przestrzeń (A^ω, d) jest zupełna.

Pokażemy teraz, że przestrzeń (A^ω, e) nie jest zupełna.

Udowodnimy najpierw, że jeśli $m > n$, to każdy NBA odróżniający słowa $a^{n!}b^\omega$ i $a^{m!}b^\omega$ ma co najmniej $f(n)$ stanów dla pewnej funkcji f, której wartości rosną do nieskończoności.

W tym celu (na mocy ograniczenia górnego na liczbę stanów DMA równoważnemu danemu NBA) wystarczy udowodnić, że każdy DMA odróżniający słowa $a^{n!}b^\omega$ i $a^{m!}b^\omega$ ma co najmniej $g(n)$ stanów dla pewnej funkcji g, której wartości rosną do nieskończoności.

Przypuśćmy, że minimalny DMA odróżniający słowa $a^{n!}b^\omega$ i $a^{m!}b^\omega$ ma nie więcej niż n stanów. Niech q_0 będzie jego stanem początkowym, natomiast q_i (dla $i \in \mathbb{N}$) - stanem po przeczytaniu słowa a^i. Wówczas na mocy zasady szufladkowej Dirichleta istnieją $0 < k < l \leq n$ takie, że $q_k = q_l$. Wobec tego dla dowolnego $i \geq 0$ zachodzi $q_{k+i} = q_{l+i}$. Ponieważ $l - k$ dzieli $m! - n!$, to także $q_{m!} = q_{n!}$. A więc $a^{n!}b^\omega$ i $a^{m!}b^\omega$ nie są odróżniane przez ten DMA. Możemy więc przyjąć $g(n) = n + 1$.

Udowodniliśmy więc, że $e(a^{n!}b^\omega, a^{m!}b^\omega) \leq \frac{1}{f(\min(n,m))}$. Zatem ciąg $w_n = a^{n!}b^\omega$ spełnia warunek Cauchy'ego. Przypuśćmy, że ma on granicę w metryce e. Wówczas na mocy (9.7) ma on tę samą granicę w metryce d. A oczywiście granicą w metryce d jest a^ω. Jednak (jak było wcześniej udowodnione) $e(w_n, a^\omega) = \frac{1}{2}$, więc a^ω nie jest granicą w metryce e, co daje sprzeczność.

4. Łatwo zauważyć, że w przestrzeni (A^ω, d) nie ma punktów izolowanych. Natomiast w przestrzeni (A^ω, e) istnieją takie. Jest nim na przykład słowo a^ω.

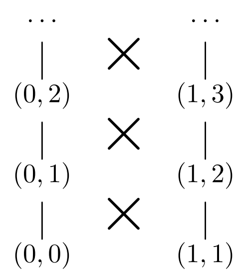
Wynika to stąd, że dla dowolnego słowa $w \neq a^\omega$ minimalny NBA odróżniający w od a^ω ma 2 stany: p (początkowy i akceptujący) i q oraz następujące przejścia:

ze stanu p po literze a jest przejście do stanu p,
ze stanu p po literze innej niż a jest przejście do stanu q,
ze stanu q po dowolnej literze jest przejście do stanu q.

Zadanie 4.6. Wykaż, że języki rozpoznawane przez automaty deterministyczne z warunkiem parzystości, używające priorytetów $\{i, \ldots, j\}$ dla $i \in \{0, 1\}$, $j \geq i$ tworzą ścisłą hierarchię.

Rozwiązanie. (Zapisał Michał Gołębiowski.)

Przypomnijmy - deterministyczny automat z warunkiem parzystości ma stany ze zbioru $\{i, i+1, \ldots, j\}$ i akceptuje słowo w wtedy i tylko wtedy, gdy największy stan (traktowany jako liczba całkowita) występujący na biegu automatu na w jest liczbą parzystą. Język rozpoznawany przez taki automat oznaczamy przez (i, j). Oczywiście można bez ograniczenia ogólności założyć, że $i \in \{0, 1\}$, gdyż języki (i, j) i $(i+2, j+2)$ są identyczne (odp. automat dla drugiego języka konstruujemy przez dodanie 2 do wszystkich stanów automatu dla pierwszego języka). Naszym celem jest pokazanie, że zachodzą zależności:



gdzie linie oznaczają <u>ścisłe</u> zawieranie „w górę".

Zawierania $(0, n) \subseteq (0, n+1)$, $(1, n) \subseteq (1, n+1)$ oraz $(1, n) \subseteq (0, n)$ są oczywiste. Zawieranie $(0, n) \subseteq (1, n+2)$ wynika z oczywistego zawierania $(2, n+2) \subseteq (1, n+2)$ oraz z równości języków $(0, n)$ i $(2, n+2)$.

Należy jeszcze wykazać ścisłość owych zawierań. Udowodnimy, że dla dowolnego $n \in \mathbb{N}$ istnieje język $L \in (1, n+1) \setminus (0, n)$. Niech naszym alfabetem będą liczby $\{1, 2, \ldots, n+1\}$. Definiujemy:

$$L = \{w \in \{1, 2, \ldots, n+1\}^\omega : \max\{n \in \mathbb{N} : \exists_i^\infty x_w(i) = n\} \text{ jest parzyste}\}$$

gdzie $x_w(i)$ to stan, w którym automat znajdzie się po przeczytaniu pierwszych i liter słowa w. Oczywiście $L \in (1, n+1)$ - wystarczy zbudować automat, który po przeczytaniu dowolnej „litery" znajdzie się w stanie o liczbie identycznej z ową „literą". Skonstruujmy indukcyjnie słowo $w \in \{1, 2, \ldots, n+1\}^\omega$. Niech $w(0)$ będzie dowolną literą. Załóżmy, że mamy już skonstruowane pierwsze i liter słowa w, niech $x(i) = k$; kładziemy wówczas $w(i+1) = k+1$. Zauważmy, że maksymalna liczba występująca nieskończenie wiele razy w słowie w jest o jeden większa od maksymalnego stanu występującego nieskończenie wiele razy na biegu automatu na w. Ponieważ jednak maksymalna liczba występująca dowolnie daleko w w jest parzysta, to maksymalny stan występujący dowolnie daleko w biegu automatu na w jest nieparzysty. W szczególności, słowo w jest nieakceptowane przez rzeczony automat, sprzeczność. Dowodzi to, że automat żaden automat z klasy $\{0, \ldots, n\}$ nie może zaakceptować języka $L \in (1, n+1)$, a zatem istnieje język $L \in (1, n+1) \setminus (0, n)$.

Analogicznie dla języka:

$$M = \{w \in \{0, 1, \ldots, n\}^\omega : \max\{n \in \mathbb{N} : \exists_i^\infty x(i) = n\} \text{ jest parzyste}\}$$

leżącego w oczywisty sposób w klasie $(0, n)$ dowodzimy, że nie jest on akceptowany przez żaden automat z klasy $\{1, \ldots, n+1\}$. Tym razem po wejściu w stan $x_w(i) = k$ kładziemy literę $w(i) = k-1$. W sposób analogiczny do poprzedniego dochodzimy znów do sprzeczności.

Udowodniliśmy zatem, że dla każdych i, j zbiory $(i, j) \setminus (i+1, j+1)$ oraz $(i+1, j+1) \setminus (i, j)$ są niepuste. Z tego, że oba te zbiory są podzbiorami $(i, j+1)$ wynika jednoznacznie, że każda z inkluzji $(i, j) \subseteq (i, j+1)$ oraz $(i+1, j) \subseteq (i, j)$ jest ścisła, QED.

Słowa skończone.

Słowa nieskończone.

Zadanie 6.14. Rozwiązać problem Parity w czasie $O(n^d)$, gdzie d to liczba różnych rang występujących w G.

Rozwiązanie. (Zapisał Oskar Skibski.)

Pokażmy najpierw jak rozwiązać problem dla dwóch rang - 1 i 2 - przy czym gracz $\exists$ wygrywa przy spełnionym warunku parzystości. Oznaczmy przez W zbiór wszystkich wierzchołków i przez W_i zbiór wierzchołków z rangą i. Obliczmy ciąg A_i następująco:

- $A_0 = W$
- $A_{i+1} = Attr^+_\exists(A_i \cap W_2)$

Łatwo zauważyć, że $A_{i+1} \subseteq A_i$. Intuicyjnie zbiór A_i oznacza zatem "wierzchołki z których gracz $\exists$ potrafi spowodować, że rozgrywka i razy wejdzie do W_2". Od pewnego momentu ciąg się ustabilizuje i dostaniemy zbiór wierzchołków, z których gracz $\exists$ potrafi spowodować, że rozgrywka wejdzie do W_2 dowolną ilość razy. Z tych wierzchołków właśnie wygrywa gracz $\exists$. Dlaczego z pozostałych wygrywa $\forall$? Opiszmy strategię tego gracza. Jeżeli gracz jest w zbiorze $A_i \setminus A_{i+1}$ to jego strategią jest unikanie zbioru $W_2 \cap A_i$. Wiemy że może to robić - gdyby nie mógł, wówczas wierzchołek ten należałby do A_{i+1}. Istnieją zatem dwie możliwości - albo $\forall$ będzie chodził tylko poza W_2 (wtedy wygra), albo wejdzie do W_2, a dokładniej $W_2 \setminus A_i$. Zejdziemy wówczas zatem do zbioru $A_j \setminus A_i$ dla $j < i$. I o to nam chodziło. Czyli gracz $\forall$ wygra wcześniej, albo po skończonej ilości jedynek zejdzie do zbioru $A_0 \setminus A_1$, czyli zbioru z którego może unikać wchodzenia do jedynek w ogóle.

Pokażemy teraz jak z gry z d rangami zrobić (maksymalnie) n gier z $d-1$ rangami dla $d > 2$. Przyjmijmy bez straty ogólności, że d jest parzyste, czyli jeżeli powtórzy się nieskończenie wiele razy wygra gracz $\exists$. Konstrukcja będzie wyglądała podobnie jak w przypadku 2 rang. Ponownie określmy ciąg zbiorów A_i, niech $A_0 = W$ i A_i będzie zbiorem wierzchołków z których gracz $\exists$ wygrywa w grze G_{i-1}. Gra G_i będzie naszym odpowiednikiem atraktora z poprzedniej definicji. Budujemy ją z G w następujący sposób:

- wierzchołki z $W_d \setminus A_i$ będą kończyły grę ze zwycięstwem $\forall$,
- wierzchołkom z $W_d \cap A_i$ zostawiamy tylko krawędzie wyjściowe i dajemy im dowolną rangę (może być 1) - pojawią się w rozgrywce maksymalnie raz,
- tworzymy sztuczną kopię wierzchołków z $W_d \cap A_i$ i zostawiamy im tylko krawędzie wchodzące - będą kończyły grę ze zwycięstwem $\exists$.

Zgodne z definicją gry "kończenie" jej w wierzchołki v ze zwycięstwem gracza j uzyskujemy przez zamianę wszystkich krawędzi wychodzących na jedną krawędź prowadzącą do niego samego i przyznanie v odpowiedniej rangi (1 lub 2) - takiej która spowoduje zwycięstwo gracza j.

Ciąg A_i jest monotonicznie malejący ($A_1 \subseteq A_0$, a potem $A_{i+1} \subseteq A_i$ równoznaczne zdaniu "G_{i+1} gorsze od G_i dla $\exists$"), zatem po pewnej ilości kroków (k) się ustabilizuje. Nasza teza jest taka, że zbiór wierzchołków jaki będzie w zbiorze A_k jest równy zbiorowi wierzchołków z których wygrywa gracz $\exists$.

Najpierw pokażmy, że wszystkie wierzchołki osiągniętego zbioru są wygrywające dla $\exists$. Rozważmy wierzchołki z $W_d \cap A_k$. Aby wygrać gracz $\exists$ musi albo dojść do wygrywającej kopii tych wierzchołków, albo wygrać niekorzystając z naszego "ułatwienia". Jeżeli potrafi wygrać nie korzystając z ułatwienia, to tak samo potrafi w zwykłej grze. Jeżeli zaś dochodzi do sztucznych wygrywających wierzchołków, to w zwykłej grze może z osiągniętej kopii grać strategią właściwego wierzchołka - z niego też wygrywa. Teraz jeżeli wierzchołki $W_d \cap A_n$ powtórzą się nieskończenie wiele razy, to $\exists$ wygra. Jeżeli nie, to znaczy że na ścieżce pojawił się wierzchołek z którego wygrywamy bez ułatwienia. To też nas urządza. Skoro mamy strategię wygrywającą z wierzchołków $W_d \cap A_n$ to wystarczy dopowiedzieć, że z pozostałych wierzchołków albo dojdziemy do nich (a właściwie ich kopii) i wygramy, albo wygramy bez nich.

Opiszmy teraz strategię gracza $\forall$ dla wszystkich wierzchołków poza A_k. Zbudujemy ją indukcyjnie. Oczywiście w zbiorze $A_0 \setminus A_1$ wygrywa $\forall$ - wygrana w G_0 oznacza wygraną w trudniejszej grze G. Rozpatrzmy zbiór $A_i \setminus A_{i+1}$. Z założenia indukcyjnego mamy już określoną strategię dla $A_0 \setminus A_i$, czyli też dla $W_d \setminus A_i$. Weźmy wierzchołek z tego zbioru - przynależność do tego zbioru oznacza, że $\forall$ wygrywa z niego w G_i i przegrywa w G_{i-1}. Rozpatrzmy jego strategię. Jeżeli wykorzystuje ona ułatwienie gracza $\forall$ (wierzchołki $W_d \setminus A_i$) to w grze G możemy grać dalej strategią określoną w założeniu indukcyjnym z tego wierzchołka. Jeżeli potrafi wygrać nie wchodząc do tego zbioru, to oczywiście strategia ta będzie możliwa do zastosowania w G.

Pokazaliśmy zatem, że zaczynając w zbiorze A_k wygrywa $\exists$, a poza nim - wygrywa $\forall$. Oznacza to, że obliczając wyniki gier G_i dostajemy podział jakiego wyznaczenie było naszym zadaniem.

Zadanie 6.16. WeakParity powstaje przez zamianę warunku parzystości na słaby warunek parzystości: największy priorytet pojawiający się choć raz jest parzysty. Pokazać, że problem WeakParity jest PTIME-zupełny (w sensie redukcji w LOGSPACE).

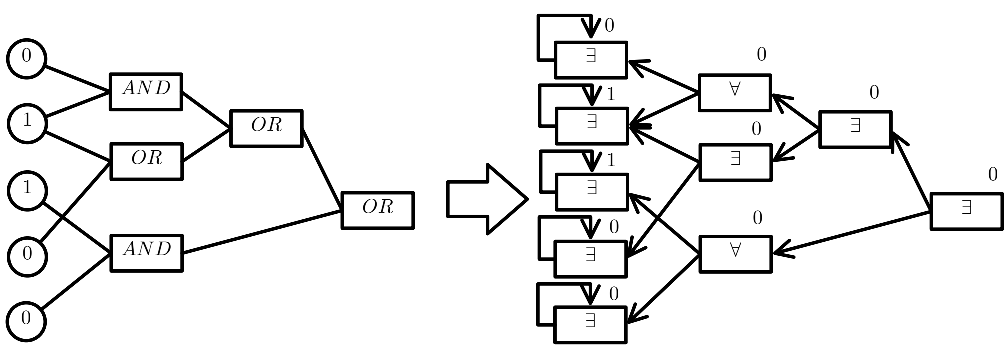
Rozwiązanie. (Zapisał Oskar Skibski.)

Pokażemy najpierw że problem ten jest rozwiązywalny w czasie wielomianowym. Niech najwyższa ranga będzie równa d i będzie zwycięska dla gracza $i \in \{\exists, \forall\}$. Policzmy zatem atraktor dla gracza i od zbioru wierzchołków z rangą d - $Attr_i(wierzchokizd)$. We wszystkich wierzchołkach atraktora oczywiście wygrywa i - może dojść do najwyższej rangi i tym sposobem wygrać. Wyrzucając z gry te wierzchołki dostajemy grę z mniejszą najwyższą rangą. Gra jest poprawna - jak każda będąca obcięciem do dopełnienia atraktora - gdyby z któregoś wierzchołka wszystkie krawędzie prowadziły do atraktora sam by w nim był. Zatem po wykonaniu maksymalnie d iteracji dostaniemy podział na wierzchołki zwycięskie gracza $\exists$ i $\forall$. Złożoność algorytmu jest oczywiście wileomianowa, możemy ją ograniczyć przez $O(n^3)$ ponieważ wykonujemy maksymalnie n iteracji obliczania atraktora, który da się policzyć w czasie $O(n^2)$.

Wiemy zatem że nasz problem jest w PTIME. Aby pokazać że problem jest PTIME-zupełny pokażemy redukcję podstawowego problemu z tej klasy - problemu obliczania wartości obwodu logicznego - do naszego problemu. Weźmy dowolny obwód logiczny złożony z bramek OR i AND. Nasza redukcja opiera się na stworzeniu odpowiedniej areny z obwodu podług zasad:

- niech każde wejście układu będzie wierzchołkiem o randze odpowiadającej wartości wejścia (0 lub 1), dowolnym właścicielu i jednej krawędzi prowadzącej do niego samego,
- niech wszystkie połączenia bramek będą krawędziami skierowanymi w kierunku wejścia,
- niech każda bramka OR będzie wierzchołkiem gracza $\exists$ o randze 0,
- niech każda bramka AND będzie wierzchołkiem gracza $\forall$ o randze 0,

przy czym gracz $\exists$ wygrywa dla rangi 1.



Łatwo zauważyć, że w naszej grze z pierwszego wierzchołka (będącego poprzednio wynikiem obwodu) wygrywa gracz $\exists$ wtedy i tylko wtedy kiedy wynikiem obwodu logicznego będzie 1. Jako że nasza redukcja jest bardzo prosta jest “przepisowa” (nie wykorzystuje zbyt dużo pamięci i działa w czasie liniowym), więc WeakParity jest PTIME-zupełny.

Drzewa skończone.

Drzewa nieskończone.